A formal approach to the modeling, simulation and analysis of nano-devices.

Sylvain Pradalier

▶ **To cite this version:**

Sylvain Pradalier. A formal approach to the modeling, simulation and analysis of nano-devices.. Bio-informatique [q-bio.QM]. Ecole Polytechnique X, 2009. Français. NNT : . tel-00780567

HAL Id: tel-00780567

https://pastel.archives-ouvertes.fr/tel-00780567

Submitted on 24 Jan 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

École Polytechnique et Università di Bologna di Scienze dell'Informazione.
Thèse de Doctorat.
Spécialité Informatique.

A formal approach to the modeling, simulation and analysis of nano-devices.

Sylvain Pradalier

Soutenue publiquement le vendredi 25 septembre 2009,
devant le jury composé de:

Vincent Danos
Pierpaolo Degano - Rapporteur
François Fages
Jérôme Feret
Jane Hillston - Rapporteur
Cosimo Laneve - Directeur
Cédric Lhoussaine
Catuscia Palamidessi - Directrice

Contents

List of symbols.

Arrows.

- $\longrightarrow$: basic transition relation, cf Section 2.1 and Definitions 3.1.5, 4.2.3 and 6.3.3.
- $\longmapsto$: collective transition relation, cf Definition 2.1.2.
- $\models\!\!\Longrightarrow$: downgraded transition relation, cf Definition 2.2.3.
- $\rightarrowtriangle$:`nano`κ rules, cf Definition 3.1.3.
- $\Longrightarrow$: weak transition relation, cf Definition 6.2.1.

Relations and equivalences.

- $\equiv$: structural congruence, cf Definitions 3.1.2, 4.2.2 and 6.6.2.
- $\overset{\circ}{\equiv}_\alpha$: restricted α-renaming, cf Section 4.5.2.
- $\sim_{IMC}$: weak markovian bisimilarity, cf Definition 2.2.4.
- $\sim_{CTMC}$: lumping equivalence, cf Definition 2.2.5.
- $\sim_{CME}$: substitution equivalence, cf Definition 5.2.3.
- $\sim_b$: backward stochastic bisimulation, cf Definition 5.3.1.
- $\sim$: master equation equivalence, cf Theorem 5.4.1.

Calculi.

- msπ: stochastic π-calculus with mixed choice and finite rates, cf Section 6.3.
- msπ^∞: stochastic π-calculus with mixed choice and infinite rates, cf Section 6.3.
- msπ^n: multi-scale π-calculus with mixed choice and rates of magnitude less or equal to n, cf Section 6.5.
- msπ^Ω: multi-scale π-calculus with mixed choice and with no bound on the magnitude of the rates, cf Section 6.5.
- ssπ: stochastic π-calculus with separate choice and finite rates, cf Section 6.3.
- ssπ^∞: stochastic π-calculus with separate choice and infinite rates, cf Section 6.3.

- ssπ^n: multi-scale π-calculus with separate choice and rates of magnitude less or equal to n, cf Section 6.5.
- ssπ^Ω: multi-scale π-calculus with separate choice and with no bound on the magnitude of the rates, cf Section 6.5.

Encodings.

- $[\![(.)]\!]$: encoding from the `nano`κ calculus to the `nano`κ calculus with gangs, cf Definition 4.3.1.
- $\{[.]\}$: encoding from the `nano`κ calculus with gangs to the `nano`π-calculus, cf Definition 4.3.2.
- $[\![.]\!]$: encoding from the probabilistic π-calculus with mixed choice to the probabilistic π-calculus with separate choice, cf Definition 6.7.1.

Symbol related to CTMC.

- λ, ρ, ν: rate of a transition, cf 2.1.1.
- μ: transition matrix, cf 2.1.1.

Symbol related to the `nano`κ calculus.

- A, B, C: species, cf Definition 3.1.1.
- S, T: solutions, cf Definition 3.1.1.
- r, s, t: fields, cf Definition 3.1.1.
- u, v, w: valuations, cf Definition 3.1.1.
- a, b, c: sites, cf Definition 3.1.1.
- ϕ, ψ, ν: interfaces, cf Definition 3.1.1.
- x, y, z: bonds, cf Definition 3.1.1.
- ϵ: the empty bond, cf Definition 3.1.1.

Symbol related to the π-calculus.

- x, y, z: channels, cf Definitions 4.2.1, 6.3.1 and 6.6.1.
- A, B, C: agents, cf Definition 4.2.1, 6.3.1 and 6.6.1.
- P, Q, R: processes, cf Definition 4.2.1, 6.3.1 and 6.6.1.
- α, β, γ: actions, cf Definition 4.2.1, 6.3.1 and 6.6.1.
- Σ or $+$: the choice operator, cf Definition 4.2.1, 6.3.1 and 6.6.1.

Chapter 1

Introduction.

Nano-muscles and nano-elevators, molecular scissors and molecular gyroscopes, molecular wheelbarrows and nano-cars, but also motors, rotors, switches, shuttles, ... Where will the imagination and ingeniousness of chemists stop in the design of nano-scale machines ? The sky is the limit. Or is it the quark?

Nano-devices. These machines operate at the nano scale and are synthesized from molecular subcomponents whose functionalities are combined in order to implement some new functionality. So they are artificial molecular complexes built on purpose to achieve a predetermined task.

For instance, rotaxanes [68] are systems composed of a dumbbell-like molecular axle surrounded by a macrocycle molecule (called "ring").

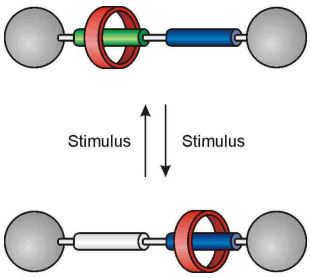


Figure 1.1: Schematic representation of a two-station rotaxane and its operation as a controllable molecular shuttle.

The extremities of the axle are bulky chemical moieties (called "stoppers") big enough to prevent the ring from escaping, which would cause the disassembly of the system. In rotaxanes containing two different recognition sites on the axle (called "stations"), it is possible to switch the position of the macrocyclic ring between the two stations by an external energy input as illustrated in Figure 1.1. Several rotaxanes of this kind, known as *molecular shuttles*, have already been developed (see [20] and the references therein) and used for building more complex systems [48, 46, 2]. The energy input necessary for the motion of the ring can be alternatively given by light, temperature, acid-base reactions or redox reactions.

There are two distinct approaches to the design of nano-devices ([35] page 163). The "hard matter" approach proposes to take inspiration from the macroscopic principles and objects and adapt them to the nano-scale. The "soft matter" approach proposes to focus on the principles of chemistry in order to design efficient, adapted and controllable motion at the molecular level. According to the authors of the above review:

> *The advantage of the "hard matter" approach is that we understand macroscopic mechanics very well and so we can easily see how to construct mechanical machines. The disadvantage is that, under most conditions, such machines must fight against the physics and*

chemistry intrinsic to their length scale. The advantage of the "soft matter" approach is that [...] we can see how to exploit the natural features of the nanoscale environment. Its disadvantage is that there is no familiar macroscopic model to follow and nature's exquisite working examples are too complex in their detail to provide us with more than broad clue at present.

[...]Both sets of design philosophies have already had many notable successes and they are not mutually exclusive. No doubt their combination will become increasingly important in the future.

In this thesis we study nano-devices stemming from the "soft matter" approach while keeping in mind that such devices could also be used in the "hard-matter" approach in the future.

Various biochemical systems have been successfully represented and studied using formalisms originating from the *Concurrency Theory*. So our goal in this thesis is to investigate a *formal representation* of nano-devices using such formalisms and the possible applications of such modelings. What is the Concurrency Theory? And what are the advantages and limitations of a formal representation using concurrent formalisms?

A formal "in-silico" laboratory. Concurrency Theory is the field of Computer Science devoted to the study of concurrent phenomena, that is processes operating in *parallel* and interacting with each other. Typical examples are web transactions, sensor networks or simply the various processes communicating inside a machine, such as a car for instance. Process algebras are a class of formalisms originating from Concurrency Theory (see for instance [55]), where processes are expressed syntactically as terms built by composing various operators. For instance, $P|Q$ is the parallel composition of the terms P and Q or $P + Q$ is the term behaving either as the term P or as the term Q. So a characteristics of the conceptual framework developed for Process algebras is the focus on structure and compositionality, in the sense that the operational semantics (which describes the possible executions of terms) is usually defined in terms of structural rules, (that is rules following the structure of the terms).

The seminal works [64, 65] have shown that biomolecular processes can be successfully represented and simulated using process algebra. This approach are grounded on the analogy between molecules and processes and between molecular reaction and process communication. It also takes advantage of the intrinsic concurrent nature of biomolecular systems: the various entities react together simultaneously and compete for resources. They are also distributed in space. Indeed, some reactions occur only on the cytogenic membrane, in the Golgi apparatus, or in the nucleus, for instance.

The advantages of this approach are manyfold. It offers a formal format for the description of biomolecular systems and this eases the exchange and gathering of models. It permits in-silico simulations, which usually saves both time and ressources with respect to in-vitro or in-vivo experiments. More importantly it permits formal analysis techniques such as model-checking, abstract

interpretation, reduction of the state space using behavioral equivalences, sensitivity analysis, etc. Finally, once one is confident with a model, one could test alternative scenario: for instance by investigating the effect of a "knock-out", that is the deletion of an interaction, or by considering alternative environmental conditions such as temperature or acidity. In brief, the goal of this approach is to provide an "in-silico" laboratory for the studying of biochemical systems.

The main limitations of this approach are the following. First, it depends on the available biomolecular data, in the sense that the agents and interactions of a model are given by the experiments, or sometimes the beliefs, of biochemists. Moreover, for the moment and to our knowledge, this approach has shown a limited predictive power, in the sense that it has not revealed many behaviors that were not experimentally observed yet. This is probably due to the lack of law of interaction at the level of abstraction of these models. Finally the analysis aspect of this approach is limited by the available computational power: the systems studied are often very large and so the resulting state space does not allow all the analysis techniques to be applied.

The systems studied in this approach are usually originating from systems biology, typically protein-protein interaction networks or regulation networks. However the arguments of the analogy molecules-processes and reactions-com munications, and of the intrinsic concurrent nature of biochemical systems, are still valid for nano-devices. And there is one more argument for applying this approach to nano-devices: their compositional nature that correspond to the compositional nature of process algebra.

Compositionality. An important feature of the "soft-matter" nano-devices is their *intrinsic compositional nature both in structure and function*. Indeed a device is built by assembling smaller molecular components and is often meant to be reused as a component for a more complex device. Moreover the function of a device is performed by combining the functions of its components. We illustrate these principles by two examples: a nano-elevator built from the assembling of three rotaxanes [2] and molecular logic circuits [73, 53].

The nano-elevator consists of the assembling of three rotaxanes and two flat tray-like molecules. One of the trays acts as a roof and has three legs, each of them consisting of a rotaxane. The other tray acts as a platform and it is interlocked with the three rings of the rotaxanes. With the appropriate stimulus it is possible to trigger the motion of the rings along the rotaxanes and obtain in this way the up and down motion of the platform of the elevator. Figure 1.2 describes the "up" and "down" configurations of the nano-elevator.

This example illustrates how nano-devices can be composed to build the structure and the function of a more sophisticated device. Maybe more importantly it emphasizes how rotaxanes are meant to be building blocks for more complex machines.

The implementation of the various features of logical circuits using molecular machines has already been investigated by chemists: in particular the basic logical gates, logical circuits constituted by a small number of logical gates

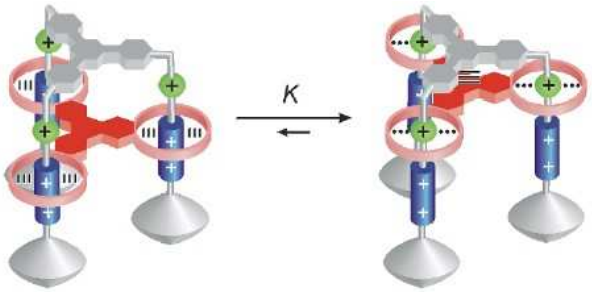


Figure 1.2: Schematic representation of the nano-elevator.

(see [73] and the references therein), as well as multiplexer and demultiplexer [53] have been implemented. Let us present the latter more in detail.

A multiplexer is a logical gate that has two boolean inputs and one boolean output. Moreover it can be in two states: in the first state the first input is directed towards the output, while in the second state the second input is directed towards the output. A demultiplexer behaves conversely: it has one boolean input and two boolean outputs and according to its state the input is directed either towards the first output or towards the second one. In order to implement these gates the authors of [53] focus on the fluorescence and light-absorption properties of the 8-Methoxyquinoline (simply written 8-MQ in the following) and its protonated form 8-MQ-H^+, as depicted in Figure 1.3[1]. The two states of both gates are embodied using 8-MQ and 8-MQ-H^+, and the input signals consist of incident light of different intensities, while the output signals consist of fluorescence intensity of different wavelength. Thresholds are chosen in order to transform these continuous signals into boolean values. The switching from 8-MQ to 8-MQ-H^+ and vice-versa is easily controlled using acid-base reactions with the triflic acid and the tris-n-butylamine respectively.

For instance in the case of the multiplexer, the input signals are light intensities of 285 nm and 350 nm while the output signal is fluorescence intensity at 474 nm. 8-MQ fluoresces with intensity at 474 nm in response to light excitation at 285 nm but not in response to excitation at 350 nm. Instead 8-MQ-H^+ fluoresces with intensity at 474 nm in response to light excitation at 350 nm but not in response to excitation at 285 nm.

Even if the wiring of the input and output signals of the various molecular gates remains a challenge, the efforts are clearly directed towards obtaining a set of basic molecular gates that can be composed in a circuit.

Formal nano-devices. The compositional and concurrent nature of nano devices indicates that formalisms originating from the Concurrency Theory and especially process algebra, which we have introduced previously, constitute ap-

[1]The Figure 1.3 has been borrowed from [53].

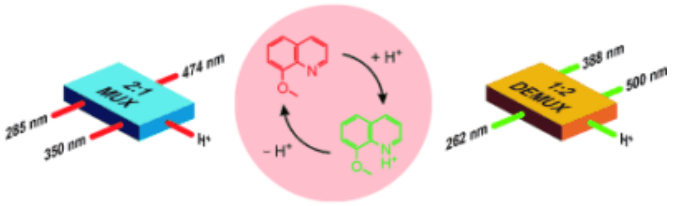


Figure 1.3: Schematic representation of 8-MQ, in red, and 8-MQ-H^+, in green, and their operations as multiplexer and as demultiplexer.

pealing candidates for the formal modeling, simulation and analysis of nano-devices.

We start with the goal of formalizing the RaH rotaxane [68] in order to simulate its behaviour *in silico* by means of some contemporary stochastic evaluator [43, 19]. What formalism should we choose ? Three main criteria drive our choice:

- First, the dynamics of chemical systems is described using the Chemical Master Equation (`CME` in the following) which provides a full description of the probabilistic behaviour of chemical systems via the dynamics of its discrete populations of molecules. It is a differential equation on $P(\rho, t)$, the probability of a stochastic system to be in the state ρ at time t, that can be derived from the Chapman-Kolmogorov equation [71]. In order to be able to reflect the `CME`, our language should be equipped with a stochastic semantics. This is also necessary to perform simulations.

- Among the various stochastic Process algebras devoted to the study of biological systems two main approaches have emerged: the *rule-based* approach [13, 5, 25, 21, 12] and the *process-oriented* approach [66, 67, 15, 43, 16]. According to the former approach – inspired by traditional chemical kinetics – a system is specified as a set of reactions; according to the latter – inspired by process calculi – a system is specified by defining each molecule as a process, and deriving the overall behaviour by means of communication rules.

  Process-oriented descriptions depart from ordinary biochemical models because they define the sequences of actions once and for all and use syntaxes usually devoted to Computer Science. Moreover the modelling of a molecule is a term of size proportional to the number of interactions addressing the molecule. As a consequence, such descriptions are less intelligible to biochemists than rule-based approaches, whose syntax is closer to biochemistry and whose complexity is spread over the reactions.

On the other hand, the theory of process-oriented calculi is much more developed and they retain several simulators and tools, which make them attractive for experiments *in silico* (see for instance [39, 15, 43, 74, 45]).

We choose to have a rule-based language in order to benefit from the nice modelling properties. However as we detail in the sequel our language can be encoded in the stochastic π-calculus [64, 65]. The π-calculus [55] is a process algebra of reference in the Concurrency Theory. It is devoted to the representation of communicating and mobile systems. Its terms model agents evolving in parallel and communicating by synchronization on channels. Its stochastic version has been successfully used for representing biochemical systems [66, 67]. Our encoding in the stochastic π-calculus permits us to also benefit from the advantages of the agent-based approach.

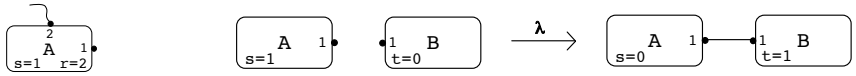
- Finally most of the structure and the function of nano-machines rely on the bound capability of the molecular components and on their change of state, typically light-induced excitation or fluorescence. So our formalism should explicitly describe these features.

Therefore a formalism such as the κ-calculus appears to be a perfect candidate. Indeed, as we detail in the following, it is a stochastic rule-based formalism that explicitly represents the bounding capabilities of the molecules.

Its basic operator is the *molecule*, which possesses sites that can be either free or bounded to others molecules sites and that can have an internal state. Its dynamic is governed by reactions that operate on molecular complexes by creating or destroying bounds and by updating the internal states. For instance, the molecule $A(s^1 + r^2 + 1 + 2^x)$ has two internal states s and r and two sites 1 and 2. The states s and r are set to 1 and 2, respectively. The molecule A is bound to another molecule on its site 2, and the bond is called x, and A is unbound on its site 1. The κ-calculus retains a graphical representation – the above molecule is rendered in Figure 1.4(a). The binary reaction

$$\rho_1 \qquad A(s^0 + 1), B(t^1 + 1) \overset{\lambda}{\rightarrow} A(s^1 + 1^x), B(t^0 + 1^x)$$

illustrated in Figure 1.4(b), specifies that every molecule A, whose internal



(a) (b)

Figure 1.4: Molecules and reactions in κ-calculus

state s equals 0 and whose site 1 is free, may react with every B, whose internal state t equals 1 and whose site 1 is free. The result is a complex where A and B are connected by a bond, called x, and the two internal states have swapped

values. The label λ of the reaction represents its rate, which corresponds to the chemical kinetic rate of the reaction.

It is worth to notice that this reaction applies to the molecule A in Figure 1(a), as well as to every other A with a different value of r and/or with an unbound site 2. That is, rules do not need to mention every internal states and site, they only constrain some of them. This essential feature permits to obtain compact modelings and to reuse them easily.

The stochastic semantics of the κ-calculus can be formulated in terms of Continuous Time Markov Chains (CTMC in the following). In this model a transition is labelled with a stochastic rate which determines its *sojourn time*, i.e. the time before the transition can be fired. The CTMCs have a rich and well-known theory which permits, in particular, the computation of the so-called *transient* and *steady states* probabilities (See for instance [70]). In addition the κ-calculus possesses an efficient simulation tool [25] and also some devoted analysis techniques such as reachability analysis and causality analysis [28, 47, 24].

The full power of the κ-calculus, however, turns out not to be necessary for the representation of nano-devices, where the kinds of interactions are simpler than the ones for which the κ-calculus is usually used (see for instance [24]). Therefore we choose to focus on a variant called the `nano`κ calculus: it corresponds to the κ-calculus restricted to binary reactions and enhanced with "exchange" reactions, which permit flipping the extremity of a bond from one molecule to another. For instance, the exchange $A(a^x), B(b) \xrightarrow{\lambda} A(a), B(b^x)$ flips the bond x from the molecule A to the molecule B. Simplicity is not the only benefit that we get from this restriction, the `nano`κ calculus can indeed be encoded in the stochastic π-calculus. The encoding enjoys a very strong correctness property: $\mathsf{S} \overset{\lambda}{\longmapsto} \mathsf{T} \Leftrightarrow [\![\mathsf{S}]\!] \overset{\lambda}{\longmapsto} [\![\mathsf{T}]\!]$, where S and T are κ-terms, λ is the rate of the transition, $[\![.]\!]$ is the encoding and $\mathsf{S} \overset{\lambda}{\longmapsto} \mathsf{T}$ is a transition between S and T whose rate is λ. So this property means that a solution and its encoding have exactly the same transitions. This permits us to benefit from the advantages of both approaches: the nice modelling aspects of the `nano`κ calculus and the rich tools and theory of the stochastic π-calculus.

Chemical Master Equation and equivalences. We are also interested in comparing the semantics of the chemical and `nano`κ models of nano-devices. The chemical semantics is the `CME` since it defines the probabilistic behaviors of a system from its kinetics description. Some important semantics of the process algebra are the various *equivalences*. These are relations defined on the set of terms that equate terms when their behaviour are "similar". Depending on the meaning of the "similarity" one gets various equivalences. It typically concerns the observable aspects of processes: the channel on which processes can communicate, the names of their outermost membranes or ambients, or in the case of the κ-family, the bound and free sites. Equivalences are useful to reason about models, for demonstrating properties and for reducing the size of a state space that one wants to study. This is particularly relevant in the case

of model-checking [69, 39].

We investigate an equivalence obtained by reasoning on the CME. It equates terms that have the same stochastic behaviors, that is the same probability over time. It is obtained by permuting and substituting in the CME the variable representing the terms equated by the equivalence.

Interestingly, it turns out that this equivalence that we defined only from the CME perspective happens to correspond to an already existing equivalence of Computer Science: *the backward stochastic bisimulation* [59, 69]. There are many variants of bisimulations but their common denominator is that they equate terms which can mimic each other's sequences of transitions. The backward bisimulation differs from the traditional bisimulations because it addresses ingoing transitions rather than outgoing ones.

A consequence on the expressiveness of the stochastic π-calculus. Interestingly our study of nano-devices bears fruits also in the garden of theoretical Computer Science and more precisely upon the expressiveness of the stochastic π-calculus.

In the field of concurrent languages, expressiveness is an important and intriguing problem. Differently from the case of sequential languages, the purpose of a program is not just to compute a function, but also to control the communication and the interaction of the various parallel components of a system. There are therefore more parameters and perspectives which must be taken into account when assessing the expressive power of a new formalism.

Most of the main process calculi proposed in literature have been widely investigated from the point of view of the expressive power, both in absolute terms, i.e. their capability to solve problems, and in relative terms, i.e. their comparison. In particular, there has been a lot of work aiming at establishing the relation between different calculi, thus providing some structure for the huge plethora of formalisms that have been proposed in the field of Concurrency. One of the goals of such investigation is, of course, to individuate languages that have the same expressive power but can be implemented in a more efficient way. The encoding itself can be valuable, as the source language, even if less efficient, may still be useful as a specification language. Another goal is to find out the constraints to implementation. For example, if a language can solve a problem that is known to be not solvable in a distributed asynchronous model (like for instance the symmetric leader election), then we know that language cannot be implemented in a totally distributed manner. The interested reader can find in [60] an extended discussion of these issues.

Surprisingly however, for an important class of calculi, the stochastic ones and the probabilistic ones, the question of relative expressiveness has not been investigated much (as far as we know), despite the fact that there have been many proposals already and that the areas are rather mature.

Here, we make a first step towards the study of relative expressiveness in the stochastic and probabilistic setting. We focus on one of the key mechanisms in Concurrency: the choice operator. This construct represents the selection

between alternative computations, and may be controlled by means of *guards*. Its importance relies on the fact that it is very useful in distributed systems for allowing processes to interact and coordinate.

One can define various kinds of choice depending on the guards that are allowed to appear in it. In process calculi, guards are usually communication actions (input and output), and it is then natural to consider the following classification:

- *input-guarded choice*: the guards can only be input actions,
- *output-guarded choice*: the guards can only be output actions,
- *separate choice*: a choice can contain input or output guards, but not both,
- *mixed choice*: a choice can contain both input and output guards,

In the classical settings it has been proved that the asynchronous π-calculus can encode input-guarded choice [58] and also separate choice [57]. On the contrary, it cannot encode mixed choice [60], that has been already proved to be strictly more powerful than the other kinds of choice [10]. In both cases, the proof of the separation result relies on the capability/incapability of expressing the solution to certain consensus problems.

We are interested in exploring whether the stochastic and probabilistic extensions of the above choice constructs presents a similar gap. In particular, we consider this question in the context of the π-calculus. Remarkably our encoding between the `nano`κ calculus and the stochastic π-calculus necessitates mixed choice, it cannot be achieved if we use only separate choices. This is the key ingredient that permits us to study the relative expressiveness of the separate choices and the mixed choice in the stochastic settings and to prove that the latter is strictly more expressive than the former.

In order to complete the picture we also investigate the relative expressiveness of the separate choices and the mixed choice in the probabilistic setting. In this setting transitions are labelled with a probability rather than a rate. In particular this means that there is no notion of time. Surprisingly it turns out that, in this settings, both kind of choices are equally expressive.

Contribution and Overview. In Chapter 2, we detail the problems specific to the derivation of a stochastic semantics and we address the specific difficulties due to the introduction of infinite rates.

In Chapter 3, we present the `nano`κ calculus. We define and illustrate its syntax and stochastic semantics. We then apply the `nano`κ calculus to describe and analyze an instance of 2-rotaxane, RaH [54, 1], for which the dynamic behaviour has been experimentally characterized in detail [36]. We consider two groups of simulations. The first ones are used to validate the model, by checking whether the experiments reproduced *in silico* coincide with those already performed *in vitro*. The second ones simulate *in silico* the expected behaviour of

the rotaxane RaH under conditions not yet observed *in vitro*. Interestingly, we show that under extreme conditions of very low concentration of rotaxane RaH, some of the assumptions usually made about the behaviour of the rotaxane in standard conditions of concentration, are no longer valid. The results of this chapter have been published in [Cre07] and in [Cre08] for the extend version.

In Chapter 4, we first illustrate the difficulties we found for defining our encoding from the `nano`κ calculus to the stochastic π-calculus. We present the syntax and stochastic semantics of the stochastic π-calculus. Then we define the two steps constituting our encoding and illustrate the encoding on a toy-modelling of a gene transcription. Finally we prove the correctness of the two steps of the encoding with respect to the stochastic semantics. The results of this chapter have been published in [Lan09].

In Chapter 5, we present the `CME`. We detail our proposal of an equivalence based on the `CME`. We then present the backward stochastic bisimulation and finally we prove the equivalence between the two equivalences. The results of this chapter are submitted for publication in [Pra09a].

In Chapter 6, we investigate the relative expressiveness of the separate choices and the mixed choice of the π-calculus in the stochastic and probabilistic setting. We first focus on the stochastic settings. We use our previous encoding from the `nano`κ calculus to the stochastic π-calculus to prove that, without infinite rates, the stochastic separate choices are strictly more expressive than the stochastic mixed choice. We then extend our result to the case of stochastic π-calculus with infinite rates. Then we show that separate choices with infinite rates can encode mixed choice without infinite rates. We also introduce the multi-scale π-calculus which extends the stochastic π-calculus with rates of several order of magnitude and lift our expressiveness results to this settings. Finally we detail the syntax and operational semantics of the probabilistic π-calculus. We present our encoding between the probabilistic mixed choice and the separate one. We also discuss the relative expressiveness of the probabilistic input and output guarded choices. The results of this Chapter concerning the probabilistic settings have been published in [Pal06], and the results concerning the stochastic settings are submitted for publication in [Pra09b].

We detail the related and future works in each section separately.

Publications.

[Cre08] A.Credi, M.Garavelli, C.Laneve, S.Pradalier, S.Silvi, and G.Zavattaro. *Modelization and simulation of nano-devices in the* `nano`κ *calculus.* Theoretical Computer Science. 408(1): 17-30 (2008). Extended version of [Cre07].

[Cre07] A.Credi, M.Garavelli, C.Laneve, S.Pradalier, S.Silvi, and G.Zavattaro. *Modelization and simulation of nano-devices in the* `nano`κ *calculus.* CMSB 2007: 168-183.

[Lan09] C.Laneve, S.Pradalier, and G.Zavattaro. *From biochemistry to stochastic processes.* In proceedings of QAPL 2009.

[Pal06] C.Palamidessi and S.Pradalier: *Expressiveness of probabilistic π-calculus.* ENTCS. 164(3): 119-136 (2006).

[Pra09a] S.Pradalier. *The connection between the* `CME` *and the backward stochastic bisimulation.* Submitted for publication.

[Pra09b] S.Pradalier. *Synchronous vs asynchronous stochastic π-calculus: a feedback from bioinformatics.* Submitted for publication

Chapter 2

Preliminaries.

In this Chapter, first in Section 2.1, we present the difficulties due to the derivation of the stochastic operational semantics, and then in Section 2.2, we address the specificities of infinite rates.

2.1 On the stochastic operational semantics.

The semantics of stochastic languages is usually given in terms of *Continuous Time Markov Chains* (CTMC in the following).

Definition 2.1.1 (Continuous time Markov chain) *A CTMC is a triple* $(S, \rightarrow, s_0)$ *where* S *is a set of states,* $s_0 \in S$ *is the initial state and* $\rightarrow$ *is a function from* $S \times S$ *to* $\mathbb{R}_{>0}$ *called the* stochastic transition relation *(or also transition matrix). We use* $s \xrightarrow{\lambda} s'$ *to represent* $(s, s', \lambda) \in \rightarrow$, λ *is called the rate.*

In this model a transition between two states s and s' is labelled with a stochastic rate λ that determines its *sojourn time*, that is the time spent in the state s before the transition could be fired. Formally the random variable governing this sojourn time is the exponential law of parameter λ: the probability of the transition being enabled within t time units is $1-e^{-\lambda \times t}$. Due to the superposition property of the exponential distribution, in a state with n outgoing transitions of rate $\lambda_1, \cdots, \lambda_n$, the probability that the sojourn time is less than t is exponentially distributed with rate $\sum_i \lambda_i$, i.e. $\mathsf{Prob}\{delay < t\} = 1-e^{-t \sum_i \times \lambda_i}$, and in this case the probability that the j-th transition is fired is $\lambda_j/(\sum_i \lambda_i)$. This is known as the *race condition*.

The operational semantics of a language is usually given in terms of a *Labelled transition system*[62]: in this approach, roughly speaking there is simply a transition from a state P to a state Q whenever P contains a redex of the language and the rewriting of this redex leads to Q. The derivation of a CTMC however is much more difficult because of the relevance of the number of redexes underlying the same transition. In order to illustrate this point let us consider a toy system where there are two species A and B and one rule $\rho : A, A \longrightarrow B, B$. This means that an occurrence of A, A can be rewritten into B, B. In the classical setting the labelled transition system contains, for instance, the following transitions:

(a) $A, A, B \rightarrow B, B, B$

(b) $A, A, A \rightarrow B, B, A$

because in both cases the left hand side of the rule is a subterm of the left solutions. If we transfer this example to the stochastic setting, the reaction is equipped with a rate λ. Then in the case of (a) we have $A, A, B \xrightarrow{\lambda} B, B, B$ and in the case of (b) we have $A, A, A \xrightarrow{3\times\lambda} B, B, A$, since there are respectively one and three occurrences of the reaction redex in (a) and (b). Formally, *the rate*

of a transition is the sum of the rates of the redexes underlying the transition. So one needs to count the number of these redexes. In contrast to the classical setting P and $P + P$ are not bisimilar in a stochastic settings.

In order to take care of the occurrences of redexes, we compute the stochastic semantics in two steps. First, we decorate the terms of the language with "tags" that permit us to distinguish between the occurrences of redexes. The derived transition relation, called the *basic transition relation*, is such that each transition corresponds to a unique redex. Then we compute the stochastic semantics by merging the transitions having the same source and target terms and by summing their rates: this is the *collective transition relation*. This approach is illustrated in the Sections 3.1, 4.2 and 6.3 with the definitions of the stochastic semantics of the `nano`κ calculus, `nano`π-calculus and stochastic π-calculus respectively.

The basic transition relation. This relation depends strongly on the syntax and the redexes of the formalism and it is often more intelligible to have an ad-hoc construction (that is less verbose in terms of tags). However it is possible to derive a general construction. Given a term T we construct its *tagged* version T^* by labelling each operator in T with a unique identifier. For instance the tagged version of the previous solution A, A, B is $A^\alpha, A^\beta, B^\gamma$: the identifier of the first A molecule is α, the identifier of the second A is β and the identifier of the B molecule is is γ.

Now the transitions of the tagged terms are derived following the traditional semantics but are additionally labelled with the set of the identifiers of all the operators constituting the occurrence of the redex and the redex itself. Having a structural operational semantics [62] is particularly helpful here. In such a semantics the derivation of the transitions follows the structure of the terms: for each n-ary operator f there is a rule which permits to lift the transitions of n subterms $t_1, \ldots, t_n$ to a transition of $f(t_1, \ldots, t_n)$. So one can additionally lift the set of the identifiers of the operators which are part of the redex. Let us illustrate this idea on the term T^*. The following derivation:

$$\dfrac{\dfrac{\dfrac{}{A \xrightarrow{\rho_L} B}\,\texttt{init}}{A, A \xrightarrow{\rho_L} B, A}\,\texttt{lift} \qquad \dfrac{}{B \xrightarrow{\rho_R} B}\,\texttt{init}}{A, A, B \xrightarrow{\lambda} B, A, B}\,\texttt{com}$$

becomes:

$$\dfrac{\dfrac{\dfrac{}{A^\alpha \xrightarrow{\rho_L}_\alpha B^\alpha}\,\texttt{init}}{A^\alpha, A^\beta \xrightarrow{\rho_L}_\alpha B^\alpha, A^\beta}\,\texttt{lift} \qquad \dfrac{}{B^\gamma \xrightarrow{\rho_R}_\gamma B^\gamma}\,\texttt{init}}{A^\alpha, A^\beta, B^\gamma \xrightarrow{\lambda}_{\alpha,\gamma} B^\alpha, A^\beta, B^\gamma}\,\texttt{com}$$

Roughly speaking the rule `init` transforms a reactant of a given rule into the corresponding product, the rule `lift` permits us to lift such a transition

to a context and the rule `com` synchronizes to complete the reaction. So the rules `init` introduces as subscripts α and γ since these are the tags of the molecules that embody the redex, the rule `lift` does not affect the subscript since it only adds the contextual solution A^β and the rule `com` makes the union of the subscripts α and γ since they are the tags of its premises. The following derivation corresponds to the occurrence of the redex embodied by A^β and B^γ, one remarks that its subscript $\{\beta, \gamma\}$ is different from the one of the previous transition: $\{\alpha, \gamma\}$.

$$\dfrac{\dfrac{\dfrac{}{A^\beta \xrightarrow{\rho_L}_\beta B^\beta}\,\texttt{init}}{A^\alpha, A^\beta \xrightarrow{\rho_L}_\beta A^\alpha, B^\beta}\,\texttt{lift} \qquad \dfrac{}{B^\gamma \xrightarrow{\rho_R}_\gamma B^\gamma}\,\texttt{init}}{A^\alpha, A^\beta, B^\gamma \xrightarrow{\lambda}_{\beta,\gamma} A^\alpha, B^\beta, B^\gamma}\,\texttt{com}$$

The structural congruence. We often do not want to distinguish between terms which differ by irrelevant syntactical details but actually represent the same solution. To cope with this issue the terms of a language are usually quotiented by a *structural congruence* [62]. An ad hoc structural congruence $\equiv$ has to be defined for each formalism and it can contain among others the associativity or commutativity of certain operators, or the α-renaming of bounded names. We choose to introduce the quotient by the structural congruence at the step of the collective transition relation, i.e. when we merge the basic transitions with the same source and target terms (see definition 2.1.2 below). So it should also be defined for the tagged terms. However as far as our computation of the basic and collective transition relations is concerned, it is sufficient to define $S^* \equiv S'^*$ if and only if $S \equiv S'$.

The collective transition relation. Once the basic transition relation is given either by the general method or by a more efficient ad hoc method, and supposing we are given a structural congruence, one can compute the collective transition relation. This is done in a parametric way: the construction depends only on the basic transition relation and the structural congruence.

Before presenting the definition of the collective transition relation we need a few notations. We refer to the terms of the language as $F, G, H, \ldots$. We refer to the basic transition relation as $\xrightarrow{\mu}_\partial$ where μ is a possible label and ∂ a set of tags.

- $next(F) = \{((\mu, \partial), G) \mid F^* \xrightarrow{\mu}_\partial G\}$;
- given a tagged term G and given a set $\mathcal{F}$ of pairs (X, H), where X is a pair of a label and a tag, and where H is a tagged term, let $[\mathcal{F}]_G$ be $\{(X, H) \mid (X, H) \in \mathcal{F} \text{ and } H \equiv G\}$;
- $can(\mathcal{F})$ is defined over sets of pairs (X, G) (the first element is a pair of a label and a tag, the second is a tagged term), such that the terms

occurring as second element of the pairs are all structurally equivalent. It returns a non-tagged term G' such that there is X with $(X, G'^*) \in \mathcal{F}$.

We can now present the formal definition of the collective transition relation:

Definition 2.1.2 (Stochastic collective transition relation) *The stochastic transition relation $\stackrel{\lambda}{\longmapsto}$ induced by a basic transition relation $\stackrel{\rho}{\longrightarrow}_\partial$ (∂ is the set of indexes labelling the transition and ρ is the redex underlying the transition) and a structural equivalence $\equiv$ on a language is the least relation satisfying the following property. Suppose that $F^* \stackrel{\rho}{\longrightarrow}_\partial G$ then:*

$$F \stackrel{\lambda}{\longmapsto} can([next(F)]_G), \text{ where}$$

$$\lambda = \sum_{((\rho,\partial),G') \in [next(F)]_G} rate(\rho)$$

The $next(.)$ function collects the outgoing basic transitions of a given term, the function $[.]_G$ selects the subset of these outgoing basic transitions that lead to a term structurally congruent to G and finally $can(.)$ selects a canonical representative among these terms.

Remarque 1 *In the case of an infinitely branching system the sum*

$$\sum_{((\rho,\partial),G') \in [next(F)]_G} rate(\rho)$$

might be infinite. These degenerated systems should be avoided by the basic transition relation since this would imply infinitely many redexes in a finite term. It would result either from an infinite number of rules having at least one occurrence in the solution or from a rule having an infinite number of occurrences in the solution. This is particularly irrelevant in the biochemical setting since solutions are finite and the possibilities of interaction also.

We prevent such phenomena by having only a finite number of rules in our modeling and checking that an occurrence of rule yields only one basic transition. An alternative would be to accept an infinite number of rules but require that at most a finite number of them apply in an given solution.

The first stochastic semantics of the π-calculus was introduced in [64]. It considered a different philosophy where the stochastic rates were not attached to redexes (i.e. channels in the π-calculus setting) but rather to actions. However a machinery similar to our basic and collective stochastic transition relation was developed in order to compute the stochastic operational semantics. The occurrences of redexes were indeed differentiated by recording their position inside the term tree.

2.2 On the infinite rates.

In our modelings of nano-devices infinite rates happen to be very useful for representing reactions which are much faster than the others. Since they might introduce non-determinism infinite rates do not fit in the format of the CTMC and their analysis techniques. Nevertheless we want to be able to reuse these models and techniques. So we first use the model of *Interactive Markov Chains* [42] (IMC in the following) to handle infinite rates and then define the *strictly Markovian* condition under which we can recover the model of CTMC.

But first we need to define the meaning of infinite rates. Intuitively if a transition has an infinite rate, it is infinitely fast: that is it takes no time to be fired. Formally the law governing the sojourn time is an exponential law of infinite parameter: for all t the probability of the transition to be enabled in t time units is $1 - e^{-\infty * t} = 1$. In particular it means that infinite rate transitions are always executed before finite rate ones.

The model of IMC extends the CTMCs with *interactive transitions* which can be output or input on a given name or silent action. These transitions are executed under the *maximal progress hypothesis* which states that interactive transitions have higher priority than regular ones and should be executed first. Thus we can represent our infinite rate transitions by silent interactive transitions. The following definition presents the IMC restricted to silent interactive transitions:

Definition 2.2.1 ((silent) Interactive Markov chain.) *An interactive Markov chain is a tuple* $(S, \longmapsto, \overset{\infty}{\longmapsto}, s_0)$ *where* S *is a set of states, where* $\longmapsto$ *is a function from* $S \times S$ *to* $\mathbb{R}_{>0}$ *and where* $\overset{\infty}{\longmapsto} \in S \times S$ *and where* $s_0 \in S$.

s_0 *is the* initial state, $\longmapsto$ *is the* stochastic transition relation *(or also transition matrix) and* $\overset{\infty}{\longmapsto}$ *is the* (silent) interactive transition relation.

Notation (transient, Markovian states). *We refer to the stochastic and interactive transition relations with only one relation: we write* $s \overset{\alpha}{\longmapsto} s'$ *either when* $\alpha = \infty$ *and* $(s, s') \in \overset{\infty}{\longmapsto}$ *or when* $\alpha \in \mathbb{R}_{>0}$ *and* $\longmapsto (s, s') = \alpha$.

A transition is immediate if it has infinite rate. We also call transient *a state which has an immediate outgoing transition and* Markovian *a state which has no immediate outgoing transition. Given a set of states* S *we note* S^t *and* S^m *the subset of its transient and Markovian states respectively.*

The derivation of a CTMC semantics by means of basic and collective transition relations can be reused in the presence of infinite rates to derive an IMC semantics. The whole procedure for the derivation of the basic transition relation can be reused as it is. For the collective transition relation the only difficulty is the definition of the sum of the rates: it suffices to define $\infty + \lambda = \infty$ for all finite rate λ.

We now carry on with the *downgrading* of an IMC into a CTMC under the *strictly Markovian* property.

The traditional approach to handle the non-determinism is to use *schedulers*. A scheduler is a function which associates to a history, that is a valid sequence of transitions, a transition continuing the history. The motivation for this approach is to consider that the non-determinism represents the impact of the environment on the system, and that a given scheduler represents a possible behavior of the environment. However the non-determinism of the biochemical systems is not due to some environment, it is quantifiable and a consequence of the stochastic competition between the possible interactions, that is the race condition in the CTMC vocabulary. Therefore we do not want to represent the non-determinism as schedulers but we want to get rid of it: it is an artifact of our use of the infinite rates.

Before presenting the downgrading of an IMC we define the *strictly Markovian* property. We write $\longmapsto^*$ for the reflexive transitive closure of the relation $\longmapsto$. A system is confluent if whenever $s \longmapsto s'$ and $s \longmapsto s''$ then there exists s''' such that $s' \longmapsto s'''$ and $s'' \longmapsto s'''$.

Definition 2.2.2 (Strictly Markovian IMC) *An IMC is* strictly-Markovian *if every subsystem consisting of silent interactive transitions is a confluent (up-to structural congruence) direct acyclic graph of finite depth.*

Intuitively this means that whenever an execution reaches a transient state s then the maximal sequences of immediate transitions starting from s terminate and converge to the same state. So each non-deterministic scenario (or scheduler) has exactly the same result on the execution. One can just choose one arbitrarily.

We first introduce the auxiliary function *next Markovian state* defined on solutions and yielding sets:

- $nextm(\mathsf{S}) = \{((\lambda, \mathsf{T}'), \mathsf{T}) \mid \mathsf{S} \stackrel{\lambda}{\longmapsto} \mathsf{T}' \stackrel{\infty}{\longmapsto}^* \mathsf{T}$ such that $\lambda \in \mathbb{R}_{>0}$, S and T Markovian$\}$

Then the downgrading of an IMC is defined by:

Definition 2.2.3 (Downgrading of IMC) *Let $(\mathcal{S}, \stackrel{\alpha}{\longmapsto}, s)$ be a strictly-Markovian IMC. The transition relation $\stackrel{\nu}{\Longmapsto}$, where $\nu \in \mathbb{R}_{>0}$, is the least one such that:*

- *For all Markovian states S and T, if $\mathsf{S} \stackrel{\lambda}{\longmapsto} \stackrel{\infty}{\longmapsto}^* \mathsf{T}$ then*

$$\mathsf{S} \stackrel{\nu}{\Longmapsto} can([nextm(\mathsf{S})]_{\mathsf{T}})$$

with

$$\nu = \sum_{((\lambda,\mathsf{T}'),\mathsf{T}'') \in [nextm(\mathsf{S})]_{\mathsf{T}}} \lambda$$

The strictly Markovian property implies that the relation $\stackrel{\nu}{\Longmapsto}$ defines a CTMC system. To assert the soundness of the downgrading process, we show in Proposition 2.2.1 below that it maps the classical semantics of IMC, the *Markovian bisimulation* [42], to the classical semantics of CTMC, the *ordinary lumping equivalence* [49].

Notation: Given a state $\mathsf{S} \in \mathcal{S}$, we write $\mathsf{S} \stackrel{\tau}{\Longrightarrow} \mathsf{S}$ if $\mathsf{S} \in \mathcal{S}^m$, we write $\mathsf{S} \stackrel{\tau}{\Longrightarrow} \mathsf{T}$ if $\mathsf{S} \in \mathcal{S}^t$ and if $\mathsf{T} = can(\{\mathsf{T}' \in \mathcal{S}^m | \mathsf{S} \stackrel{\infty}{\longmapsto}{}^* \mathsf{T}'\})$. Given a Markovian state S and a set of states $C \subseteq \mathcal{S}$, we denote with $\mu(\mathsf{S}, C)$ the *cumulative rate* obtained as the sum of all rates of the transitions from S to a state in C. This is formally defined as follows:

$$\mu(\mathsf{S}, C) \ = \sum_{\mathsf{S} \stackrel{\lambda}{\longmapsto} \mathsf{T}, \mathsf{T} \in C} \lambda$$

Before presenting the definition of the Markovian bisimulation and of the lumping equivalence, we use the introduced notation to state the following lemma.

Lemma 2.2.1 *Consider a strictly-Markovian IMC* $(\mathcal{S}, \stackrel{\lambda}{\longmapsto})$*, two Markovian states* S *and* T*, and the set of states* $C = \{\mathsf{S}' | \mathsf{S}' \stackrel{\tau}{\Longrightarrow} \mathsf{T}', \mathsf{T}' \equiv \mathsf{T}\} \cup \{\mathsf{T}' | \mathsf{T}' \equiv \mathsf{T}\}$*, we have that:*

- $\mu(\mathsf{S}, C) > 0$ *if and only if there exists one and only one* $\mathsf{T}'' \equiv \mathsf{T}$ *such that* $\mathsf{S} \stackrel{\mu(\mathsf{S},C)}{\Longmapsto} \mathsf{T}''$*.*

Proof The statement is a direct consequence of the Definition 2.2.3.□

Remarque 2 *The above lemma has the following implications:*

- *the probability distribution of the sojourn time in a Markovian state is the same in the IMC and in the downgraded CTMC and*
- *the probability that one of the paths* $\mathsf{S} \stackrel{\lambda}{\longmapsto} \stackrel{\infty}{\longmapsto}{}^* \mathsf{T}'$ *with* $\mathsf{T}' \equiv \mathsf{T}$ *is taken in the IMC corresponds to the probability that the unique transition* $\mathsf{S} \stackrel{\lambda'}{\Longmapsto} \mathsf{T}''$*, with* $\mathsf{T}'' \equiv \mathsf{T}$*, is taken in the downgraded CTMC.*

In IMCs, the Markovian bisimulation is built from the classical concepts of bisimulation corresponding to its interactive and Markovian parts. Two bisimilar transient states should have the same outgoing interactive transitions. Two bisimilar Markovian states should have same outgoing rates to the bisimulation equivalence classes. It is naturally extended to the notion of *weak Markovian bisimulation*. We formally detail this notion for strictly-Markovian IMCs as follows.

Definition 2.2.4 (Weak Markovian bisimulation) *Given a strictly-Markovian IMC* $(\mathcal{S}, \stackrel{\lambda}{\longmapsto})$*, an equivalence relation* $\mathcal{R}$ *on* $\mathcal{S}$ *is a* weak Markovian bisimulation *if* $\mathsf{S}\mathcal{R}\mathsf{S}'$ *implies that:*

- *if* $\mathsf{S} \stackrel{\tau}{\Longrightarrow} \mathsf{T}$ *then there exists* T' *such that* $\mathsf{S}' \stackrel{\tau}{\Longrightarrow} \mathsf{T}'$ *and* $\mathsf{T}\mathcal{R}\mathsf{T}'$*;*
- *and for all* $\mathcal{R}$*-equivalence classes* C *we have that* $\mu(\mathsf{S}, C) = \mu(\mathsf{S}', C)$*.*

Two states S *and* S' *are bisimilar if* $\mathsf{S}\mathcal{R}\mathsf{S}'$ *for some weak Markovian bisimulation* $\mathcal{R}$*. We write* $\mathsf{S} \sim_{IMC} \mathsf{S}'$*.*

Performing the ordinary lumping of a CTMC consists of agglomerating states that have equivalent forward behavior, that is from which the outgoing rates to agglomerated states are equal.

Definition 2.2.5 ((Ordinary) Lumping equivalence) *Given a CTMC* $(\mathcal{S}, \stackrel{\lambda}{\longmapsto})$*, a partitioning* $\mathcal{P}$ *of* $\mathcal{S}$ *is a* lumping *if for every pair of partitions* C *and* C' *(i.e.* $C, C' \in \mathcal{P}$*) we have that*

- *if* $\mathsf{S}, \mathsf{T} \in C$ *then* $\mu(\mathsf{S}, C') = \mu(\mathsf{T}, C')$.

Two states S *and* T *are lumping-equivalent if they are contained in the same partition of a lumping. We write* $\mathsf{S} \sim_{CTMC} \mathsf{T}$.

The soundness of the downgrading process is formalized as follows: two Markovian states in a strictly-Markovian IMC are bisimilar if and only if they are lumping-equivalent in the corresponding downgrading.

Proposition 2.2.1 *Given a strictly-Markovian IMC* $(\mathcal{S}, \stackrel{\lambda}{\longmapsto})$*, its downgrading* $(\mathcal{S}^m, \stackrel{\lambda}{\Longmapsto})$*, and two Markovian states* $\mathsf{S}, \mathsf{T} \in \mathcal{S}^m$*, we have that:*

- $\mathsf{S} \sim_{IMC} \mathsf{T}$ *if and only if* $\mathsf{S} \sim_{CTMC} \mathsf{T}$.

Proof We first consider the *only if* part. If $\mathsf{S} \sim_{\mathrm{IMC}} \mathsf{T}$ we prove that the partitioning of $\mathcal{S}^m$ made by the equivalence classes of $\sim_{\mathrm{IMC}}$ is a lumping, which implies that $\mathsf{S} \sim_{\mathrm{CTMC}} \mathsf{T}$ because S and T belong to the same equivalence class. Indeed, given two states S' and T' belonging to the same element of the partition (and so Markovian), and given any element C of the partition let $\overline{C}$ be the $\sim_{\mathrm{IMC}}$-equivalence class including C, and let μ_i and μ_c be the cumulative rates on the IMC $(\mathcal{S}, \stackrel{\lambda}{\longmapsto})$ and its downgrading $(\mathcal{S}^m, \stackrel{\lambda}{\Longmapsto})$, respectively. We have:

$$
\begin{array}{rcll}
\mu_c(\mathsf{S}', C) & = & \displaystyle\sum_{(\lambda,\mathsf{Z}) \in \{(\lambda,\mathsf{Z}) \mid \mathsf{S}' \stackrel{\lambda}{\Longmapsto} \mathsf{Z}, \mathsf{Z} \in C\}} \lambda & \\
& = & \mu_i(\mathsf{S}', \overline{C}) & \text{by Lemma 2.2.1} \\
& = & \mu_i(\mathsf{T}', \overline{C}) & \text{because } \mathsf{S}' \text{ and } \mathsf{T}' \text{ are bisimilar} \\
& = & \displaystyle\sum_{(\lambda,\mathsf{Z}) \in \{(\lambda,\mathsf{Z}) \mid \mathsf{T}' \stackrel{\lambda}{\Longmapsto} \mathsf{Z}, \mathsf{Z} \in C\}} \lambda & \text{by Lemma 2.2.1} \\
& = & \mu_c(\mathsf{T}', C) &
\end{array}
$$

We now consider the *if* part. If $\mathsf{S} \sim_{\mathrm{CTMC}} \mathsf{T}$ then there exists a partitioning $\mathcal{P}$ such that S and T belong to the same partition. It is not restrictive to assume that all partitions of $\mathcal{P}$ are closed under structural congruence (i.e. if W and Z belong to the same partition, then also W' and Z' belong to the same partition if $\mathsf{W} \equiv \mathsf{W}'$, and $\mathsf{Z} \equiv \mathsf{Z}'$). We show how to define an equivalence relation $\mathcal{R}$ on $\mathcal{S}$ which is a weak Markovian bisimulation and such that $\mathsf{S} \mathcal{R} \mathsf{T}$ (which will imply $\mathsf{S} \sim_{\mathrm{IMC}} \mathsf{T}$). We define $\mathcal{R}$ such that $\mathsf{Q} \mathcal{R} \mathsf{Q}'$ if and only if:

- Q and Q' are Markovian and belong to the same partition of $\mathcal{P}$ or

- Q and Q' are transients and there exists R and R' belonging to a partition of $\mathcal{P}$ such that $\mathsf{Q} \stackrel{\tau}{\Longrightarrow} \mathsf{R}$ and $\mathsf{Q}' \stackrel{\tau}{\Longrightarrow} \mathsf{R}'$.

Because of the convergence property of strictly Markovian IMCs and because the partitions of $\mathcal{P}$ are closed under structural congruence, $\mathcal{R}$ is an equivalence relation. Moreover, each $\mathcal{R}$-equivalence class C is composed of the Markovian states of a partition of $\mathcal{P}$, that we denote with $\overline{C}$, plus the transient states Q such that $\mathsf{Q} \stackrel{\tau}{\Longrightarrow} \mathsf{Q}'$ for some $\mathsf{Q}' \in \overline{C}$. As S and T belong to the same partition of $\mathcal{P}$, then $\mathsf{S}\mathcal{R}\mathsf{T}$.

We complete the proof showing that $\mathcal{R}$ is a weak Markovian bisimulation. Consider $\mathsf{Q}\mathcal{R}\mathsf{Q}'$ and an $\mathcal{R}$-equivalence class C (and its corresponding partition $\overline{C}$ of $\mathcal{P}$). As we have already observed in the previous paragraph, the equivalence classes are closed under immediate transitions so if $Q \stackrel{\tau}{\Longrightarrow} R$ then $\exists R'.\ Q' \stackrel{\tau}{\Longrightarrow} R'$ and $R\mathcal{R}R'$. Otherwise Q and Q' are Markovian and we can apply Lemma 2.2.1 as follows (μ and $\overline{\mu}$ are the cumulative rates on the IMC $(\mathcal{S}, \stackrel{\lambda}{\longmapsto})$ and its downgrading $(\mathcal{S}^m, \stackrel{\lambda}{\models\!\Longrightarrow})$, respectively):

$$\begin{array}{rcll}
\mu(\mathsf{Q}, C) & = & \sum_{(\lambda,\mathsf{Z}) \in \{(\lambda,\mathsf{Z}) \mid \mathsf{Q} \stackrel{\lambda}{\models\!\Longrightarrow} \mathsf{Z}, \mathsf{Z} \in \overline{C}\}} \lambda & \text{by Lemma 2.2.1} \\
& = & \overline{\mu}(\mathsf{Q}, \overline{C}) & \\
& = & \overline{\mu}(\mathsf{Q}', \overline{C}) & \text{because } \mathsf{Q} \text{ and } \mathsf{Q}' \text{ belong to the same partition} \\
& = & \sum_{(\lambda,\mathsf{Z}) \in \{(\lambda,\mathsf{Z}) \mid \mathsf{Q}' \stackrel{\lambda}{\models\!\Longrightarrow} \mathsf{Z}, \mathsf{Z} \in \overline{C}\}} \lambda & \\
& = & \mu(\mathsf{Q}', C) & \text{by Lemma 2.2.1}
\end{array}$$

□

Chapter 3

A formal language for nano-devices.

In this Chapter we investigate the formal description of nano-devices. As detailed in the Chapter 1, the κ-calculus[26] constitutes a good candidate because it has a stochastic semantics, it is rule-based and it permits us to represent explicitly sites and internal states. Moreover it has an efficient simulator and some causality and reachability analysis techniques. However the description of nano-devices does not require the full power of the κ-calculus. So we focus on the study the nanoκ calculus. It is simple and adequate for the description of nano-devices and it retains the good properties of the κ-calculus. Moreover it can also be encoded in the stochastic π-calculus, which permits us to reuse its tools and theory, as we will see in the next Chapter.

The Chapter is organized as follows. In Section 3.1, we introduce the syntax and semantics of the nanoκ calculus. In Section 3.2, we present a modeling of the rotaxane in the nanoκ calculus and present several simulations. The Chapter is closed by a conclusion in Section 3.3 and a discussion on related works in Section 3.4.

3.1 The nanoκ calculus: syntax and semantics.

Definition 3.1.1 (nanoκ solutions, nanoκ pre-solutions) *The* nanoκ *calculus uses several sets of names:* species *ranged over by A, B, C, ...,* fields *names ranged over by r, s, t, ...,* sites *ranged over by a, b, c, ..., and* bonds *names that are* totally ordered and countable *and ranged over by x, y, z, ... In order to reflect their biochemical meaning, species, fields and sites are often addressed using strings of characters.*

We also suppose given three functions $\mathfrak{s}_f(.)$, $\mathfrak{f}(.,.)$ and $\mathfrak{s}_s(.)$: $\mathfrak{s}_f(.)$ associates to each species a set of fields, $\mathfrak{f}(.,.)$ associates to each species and fields pair a finite set of integers, and $\mathfrak{s}_s(.)$ associates to each species a set of sites.

A valuation *of a species A is a function, possibly partial, which maps the fields $r \in \mathfrak{s}_f(A)$ to a value in $\mathfrak{f}(A, r)$. Valuations are ranged over by u, v, w, ... An interface of a species A is an injective map, possibly partial, from $\mathfrak{s}_s(A)$ to either bonds or a special value ε. Interfaces are ranged over by σ, ϕ, ν, ...*

The terms defined by the following grammar:

$$\mathsf{S} \quad ::= A[u](\sigma) \;\mid\; \mathsf{S},\mathsf{S} \;\mid\; \_$$

are called solutions *when all the maps are total and* pre-solutions *otherwise. The terms $A[u](\sigma)$ are called molecules. _ is the empty solution. The operator "," is assumed to be associative, i.e. $(\mathsf{S},\mathsf{T}),\mathsf{R}$ is equal to $\mathsf{S},(\mathsf{T},\mathsf{R})$ and therefore parentheses are always omitted.*

Bonds always occur at most twice in solutions. A solution or a pre-solution is proper *if every bond therein occurs exactly twice.*

Intuitively, a molecule $A[u](\sigma)$ is determined by the species A to which it belongs, and its valuation u and its interface σ. The values of the fields in u represent the internal state of the molecule, for instance its electronic charges, or some missing or additional protons. The sites in the interface σ represent

the binding capabilities of the molecule. A site a mapped to a bond name x means that a bond, called x, is established between a and a site of some other molecule. A site mapped to the special value ε is free, it is not involved in any bond.

For example, if the species A has two fields r and *phos* and three sites nh, *bipy*, and 3 the following term is a molecule: $A[r \mapsto 0; phos \mapsto 1](nh \mapsto \varepsilon; bipy \mapsto x;)$. The fields r and *phos* have values 0 and 1, respectively; the site nh is free, the site *bipy* is bond and the bond is x and this interface does not define the state of the site 3, which may be bond or not.

Notation. In order to ease the reading, we write this molecule as $A[r^0 + phos^1](nh + bipy^x)$ (the value ε is always omitted). Let $\emptyset$ be the empty map. We write $A(\sigma)$ instead of $A[\emptyset](\sigma)$, $A[u]$ instead of $A[u](\emptyset)$, and simply A instead of $A[\emptyset](\emptyset)$. We denote by $\mathrm{ran}(\sigma)$ the range of an interface σ omitting ε and by $bonds(\mathsf{S})$ the set of bonds appearing in the solution S.

Remarque 3 *We require the set of bond names to be totally ordered in order to ease the building of a finitely branching basic transition relation (see 3.1.5). The countability of the set of bond names is used only for the encoding into the* nanoπ*-calculus (see the next Chapter and in particular the definition 4.3.1).*

Example 3.1.1 *As a running example we consider a toy chemical reaction:*

$$AB \longleftrightarrow A^+ + B^-$$

In these reactions, the complex formed by the two molecules of the species A and B can be dissociated in the two ions A^+ and B^-, and vice versa. The molecules of the two species can be in two possible states: either they have a positive charge, i.e. a missing electron, like A^+, or a negative charge, i.e. an additional electron, like B^- or they are in their standard states A and B. We model these possible states using one field e with values -1, 0 and 1 that denote respectively an additional electron, no missing or additional electron and a missing electron. Moreover we model the possible complexation using a site called bond. Formally we can use $A[e^1](bond)$ for A^+, $B[e^{-1}](bond)$ for B^- and $A[e^0](bond^x), B[e^0](bond^x)$ for AB respectively.

The structural congruence of the nanoκ calculus is given by the following definition:

Definition 3.1.2 (Structural congruence of nanoκ) *The structural equivalence between solutions, denoted $\equiv$, is the least congruence satisfying the following three rules (we recall that solutions are already quotiented by associativity of ","):*

1. $\mathsf{S}, \mathsf{T} \equiv \mathsf{T}, \mathsf{S}$;
2. $\_, \mathsf{S} \equiv \mathsf{S}$;

3. $\mathsf{S} \equiv \mathsf{T}$ *if there exists an injective renaming* $\imath$ *on bonds such that* $\mathsf{S} = \imath(\mathsf{T})$.

Example 3.1.2 *Commutativity and injective renaming of the structural equivalence make it possible to prove:*

$$A[e^0](bond^x), B[e^0](bond^x) \;\equiv\; B[e^0](bond^z), A[e^0](bond^z)$$

The dynamics of the nanoκ calculus is governed by means of reaction rules. These rules correspond closely to the biochemical reactions we wish to model. Intuitively a nanoκ term can perform a transition when it contains an instance of the left hand side of a rule. Before formally presenting the rules of the nanoκ calculus a few preliminary definitions are in order:

- we write $\sigma \leq \sigma'$ if $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$ and, for every i, if $\sigma(i) \neq \varepsilon$ then $\sigma(i) = \sigma'(i)$ (intuitively all the bonds present in σ appear also in σ');
- when we write $u + u'$ and $\sigma + \sigma'$ we assume that $\mathrm{dom}(u) \cap \mathrm{dom}(u') = \emptyset$ and $\mathrm{dom}(\sigma) \cap \mathrm{dom}(\sigma') = \emptyset$.

We can now define the different kinds of rules for the nanoκ calculus:

Definition 3.1.3 *Reactions of* nanoκ *calculus are either* creations, destructions, *or* exchanges *and they are labelled by a* rate, *which is a positive real number or* ∞. *Creations have the format:*

$$A[u](\sigma), B[v](\tau) \overset{\lambda}{\rightarrow} \quad A[u'](\sigma'), B[v'](\tau'), C_1[w_1](\eta_1), \cdots, C_n[w_n](\eta_n)$$

where both hand sides are proper pre-solutions and where $\sigma \leq \sigma'$, $\tau \leq \tau'$, $\mathrm{dom}(u) = \mathrm{dom}(u')$, $\mathrm{dom}(v) = \mathrm{dom}(v')$, *and* w_i *and* η_i *are total. Destructions have one of the formats:*

$$A[u](\sigma), B[v](\tau) \overset{\lambda}{\rightarrow} A[u'](\sigma'), B[v'](\tau')$$
$$A[u](\sigma), B[v](\tau) \overset{\lambda}{\rightarrow} A[u'](\sigma')$$

where both hand sides are proper pre-solutions and where $\sigma \geq \sigma'$, $\mathrm{dom}(u) = \mathrm{dom}(u')$, *and, in the first case,* $\tau \geq \tau'$, $\mathrm{dom}(v) = \mathrm{dom}(v')$ *and, in the second case,* τ *has to be total. Exchanges have one of the formats:*

$$A[u](\sigma), B[v](\tau) \overset{\lambda}{\rightarrow} A[u'](\sigma), B[v'](\tau)$$
$$A[u](a^x + \sigma), B[v](b + \tau) \overset{\lambda}{\rightarrow} A[u'](a + \sigma), B[v'](b^x + \tau)$$

where the pre-solutions $A[u](\sigma), B[v](\tau)$ *and* $A[u](a+\sigma), B[v](b+\tau)$ *are proper and* $\mathrm{dom}(u) = \mathrm{dom}(u')$ *and* $\mathrm{dom}(v) = \mathrm{dom}(v')$.

In the rest of the thesis we assume that reactants share at most one bond, i.e. $\mathrm{ran}(\sigma) \cap \mathrm{ran}(\rho)$ *is either an empty set or a singleton.*

Creations produce new bonds between two unbound sites and/or synthesize new molecules. Destructions behave in the other way around. Exchanges either leave the interfaces unchanged or move one bond from a reactant to the other, which we call bond-flipping exchange.[1]

It is worthwhile to remark that reactions do not address every field and site of the reactants (both hand sides of a rule are pre-solutions). The intended meaning is that two molecules react if they are *instances* of the left-hand side of a reaction. We will formalize this notion later on in the basic transition relation (see definition 3.1.5).

Example 3.1.3 *The* nanoκ *calculus reactions that corresponds to the two reactions of our toy example are:*

$$A[e^0](bond^x), B[e^0](bond^x) \overset{100}{\rightarrow} A[e^1](bond), B[e^{-1}](bond)$$
$$A[e^1](bond), B[e^{-1}](bond) \overset{10}{\rightarrow} A[e^0](bond^x), B[e^0](bond^x)$$

where we have considered a rate 100 *for the left to right direction and* 10 *for the right to left direction.*

Stochastic semantics of the nanoκ calculus. We can now present the stochastic semantics of nanoκ. As we anticipated in the Chapter 2, it is achieved in several steps: first we build the basic transition relation, then the collective transition relation is derived from the basic one and finally the resulting IMC is downgraded into a CTMC, assuming that our system meets the strictly Markovian property.

The semantics depends strongly on the sets of species and of reactions considered. We formalize this with the notion of nanoκ *system*:

Definition 3.1.4 (nanoκ systems) *A* nanoκ *system is a tuple determined by* $\mathcal{S}$ *a set of species names,* $\mathcal{N}$ *a set of fields and sites names,* $\mathcal{B}$ *a totally ordered and countable set of bond names,* $\mathfrak{s}_f(.)$ *a map yielding the fields of a species,* $\mathfrak{f}(.,.)$ *a map yielding the set of possible values of a field of a species,* $\mathfrak{s}_s(.)$ *a map yielding the sites of a species and* $\mathcal{R}$ *a set of reactions.*

Notation. *We refer to a* nanoκ *system as* $(\mathcal{S},\mathcal{R})$ *and keep the other elements implicit in* $(\mathcal{S},\mathcal{R})$*.*

We now present the basic transition relation of the nanoκ calculus. In this case it is not necessary to follow the general methods presented in the Chapter 2, there exists a more efficient ad hoc method. Indeed since a nanoκ solution can be seen as a sequence of molecules, the redex of a rule is uniquely identified by the position of the two reactants inside this sequence. Therefore we only use pairs of integers as identifiers. Note however that this process is much eased

[1] The terms creation and destruction have been preferred to *complexation* and *decomplexation* used in [26, 52] because they have a more neutral chemical meaning.

by postponing the structural congruence to the step of the collective transition relation.

The definition of the basic transition relation of the nanoκ calculus requires some notation. Let μ range over $\rho_L, \imath$ and $\rho_R, \imath$ and let $\overline{\rho_L, \imath} = \rho_R, \imath$ and $\overline{\rho_R, \imath} = \rho_L, \imath$ where $\imath$ is an injective renaming (notice that $\overline{\overline{\mu}} = \mu$). The nano$\kappa$ reactions may be addressed by:

$$A[u](\sigma), B[v](\rho) \overset{\lambda}{\twoheadrightarrow} A[u'](\sigma'), \mathsf{S}$$

where S may also be $\_$. With an abuse of notation we lift a renaming $\imath$ to a solution by applying it pointwise. Finally we denote the set of names present in a solution $\mathtt{S}$ with $\mathrm{name}(S)$.

Definition 3.1.5 *Given a* nanoκ *system whose set of reactions is $\mathcal{R}$, its basic transition relation, written either $\xrightarrow{\rho,\imath}_{\ell,\ell'}$ or $\xrightarrow{\mu,\imath}_{\ell}$, is the least relation that satisfies the following rules:*

- *(init) If $\rho = A[u](\sigma), B[v](\phi) \overset{\lambda}{\twoheadrightarrow} A[u'](\sigma'), \mathsf{S} \in \mathcal{R}$, then for all ν we have both:*
  - $A[u + w](\imath \circ \sigma + \nu) \xrightarrow{\rho_L,\imath}_1 A[u' + w](\imath \circ \sigma' + \nu)$ *and*
  - $B[v + w](\imath \circ \phi + \nu) \xrightarrow{\rho_R,\imath}_1 \mathsf{T}$

  where T is either $B[v' + w](\imath \circ \phi' + \nu), \imath(\mathsf{S}')$ if $\mathsf{S} = B[v'](\phi'), \mathsf{S}'$ or $\imath(\mathsf{S})$ otherwise and where $\imath$ is an order-preserving injective renaming with $\mathrm{ran}(\imath) \cap \mathrm{ran}(\nu) = \emptyset$;

- *(lift) if $\mathsf{S} \xrightarrow{\mu,\imath}_{\ell} \mathsf{S}'$ then both:*
  - $\mathsf{S}, \mathsf{T} \xrightarrow{\mu,\imath}_{\ell} \mathsf{S}', \mathsf{T}$ *and*
  - $\mathsf{T}, \mathsf{S} \xrightarrow{\mu,\imath}_{\ell' + \ell} \mathsf{T}, \mathsf{S}'$

  where T has ℓ' molecules and where $(\mathrm{name}(\mathsf{S}') \setminus \mathrm{name}(\mathsf{S})) \cap \mathrm{name}(\mathsf{T}) = \emptyset$ if the rule of μ is a creation;

- *(communication) if $\mathsf{S} \xrightarrow{\mu,\imath}_{\ell} \mathsf{S}'$ and $\mathsf{T} \xrightarrow{\overline{\mu},\imath}_{\ell'} \mathsf{T}'$, let ρ be the rule of μ and let $\jmath$ be an order-preserving injective renaming which maps $\mathrm{name}(\mathsf{S}', \mathsf{T}') \setminus \mathrm{name}(\mathsf{S}, \mathsf{T})$ (i.e. the created names) into the least bonds not belonging to $\mathrm{name}(\mathsf{S}, \mathsf{T})$ then:*
  - $\mathsf{S}, \mathsf{T} \xrightarrow{\rho,\jmath}_{\ell,\ell'' + \ell'} \jmath(\mathsf{S}', \mathsf{T}')$

  where S has ℓ'' molecules.

The tags of the basic transition relation of the nanoκ calculus are integers or pairs of integers. According to the general approach presented in the preliminaries one would have to label each molecule with a unique identifier which would be used as subscript when the molecule is part of a redex. However in our case it is sufficient to record the position of the molecule inside the sequence of molecules. For instance supposing that the reaction:

$$A[e^1](bond), B[e^{-1}](bond) \twoheadrightarrow A[e^0](bond^y), B[e^0](bond^y)$$

is labelled ρ, then the solution $A[e^1](bond), A[e^1](bond), A[e^1](bond), B[e^{-1}](bond)$ has three outgoing basic transitions:

$$\begin{array}{l} A[e^1](bond), A[e^1](bond), A[e^1](bond), B[e^{-1}](bond) \xrightarrow{\rho}_{1,4} \\ \quad A[e^0](bond^y), A[e^1](bond), A[e^1](bond), B[e^0](bond^y) \\ A[e^1](bond), A[e^1](bond), A[e^1](bond), B[e^{-1}](bond) \xrightarrow{\rho}_{2,4} \\ \quad A[e^1](bond), A[e^0](bond^y), A[e^1](bond), B[e^0](bond^y) \\ A[e^1](bond), A[e^1](bond), A[e^1](bond), B[e^{-1}](bond) \xrightarrow{\rho}_{3,4} \\ \quad A[e^1](bond), A[e^1](bond), A[e^0](bond^y), B[e^0](bond^y) \end{array}$$

The basic transition relation uses also finite injective renamings. We first present the role of the $\imath$ renaming in the (init) rule and then the role of the $\jmath$ renaming in the (communication) rule.

The role of the renaming of the (init) rule is to allow the instantiation of the bond names of a rule in a given solution. To clarify this point, consider the creation $\varrho' = \ C(1^x + 2), C(1^x + 2) \overset{10}{\twoheadrightarrow} C(1^x + 2^y), C(1^x + 2^y)$ (a bond is created between two C molecules provided they are already bond). Then take the solution $C(1^z+2), C(1^v+2), C(1^z+2), \ C(1^v+2)$. We derive the expected transition

$$\begin{array}{l} C(1^z + 2), C(1^v + 2), C(1^z + 2), C(1^v + 2) \\ \xrightarrow{\varrho'}_{1,3} \ C(1^z + 2^w), C(1^v + 2), C(1^z + 2^w), \ C(1^v + 2) \end{array}$$

following a structured operational semantics approach [62]. Namely, we focus on the single reactants and lift the transitions to "," -contexts. This is correct to the extent that one records the instantiation of bonds in the left-hand sides of reactions with the actual names of the molecules: the two reactants must instantiate bonds in the same way. This is the reason why the first two molecules of the above solution cannot react with ϱ. More precisely, $C(1^z + 2) \xrightarrow{\varrho'_L, \imath}_1 C(1^z + 2^w)$, where $\imath = [x \mapsto z, y \mapsto w]$, and $C(1^v + 2) \not\xrightarrow{\varrho'_R, \imath}_1$.

The role of the renaming in the (communication) rule is to ensure that for a given a reaction and a pair of molecules of a given solution, one can derive at most one basic transition corresponding to these molecules and this reaction. If we do not require that the renaming is injective and order-preserving we would be able to derive a transition $C(1^x+2), C(1^x+2) \xrightarrow{\varrho'}_{1,2} \ C(1^x+2^y), C(1^x+2^y)$ for any free name y and so one occurrence of one redex would yield infinitely many basic transitions.

Thus we need to choose one transition among these possibilities. By asking that the created are the least ones we prevent the infinite number of possible transitions. However since several bonds can be created by a reaction this only ensures that the number of possible transitions is finite but not equal to 1. So we also ask that the renaming is order preserving. This permit us to choose one transition: the one where the name of the least created bond is mapped to the

least name not present in the solution, the name of second least created bond to the second least name not present in the solution, ...

It is also worthwhile to notice that there is no rule lifting a transition $\xrightarrow{\mu}_{\ell,\ell'}$ to a context ",": we use the associativity of , to partition a solution S into S',S'' such that the reactants are in S' and S''.

Remarque 4 *One might wish to derive transition constituted of the firing of several reactions. This approach seems relevant since all the reactions happen in parallel. However the Gillespie algorithm [37], which is the standard simulation method for stochastic process algebra, is not compatible with this approach. Indeed the Gillespie algorithm simulates systems of biochemical reactions by probabilistically selecting the next reaction to happen and the time spent before it happens.*

The compatibility of the structural congruence with respect to the basic transition relation is stated in the following proposition:

Proposition 3.1.1 *Let* $\mathsf{S} \equiv \mathsf{S}'$.

1. *If* $\mathsf{S} \xrightarrow{\mu}_{\ell} \mathsf{T}$ *then there exists a* T' *and a renaming* $\imath$ *such that* $\mathsf{S}' \xrightarrow{\imath(\mu)}_{\ell'} \mathsf{T}'$ *and* $\mathsf{T} \equiv \mathsf{T}'$ *(with* $\imath(\mu)$ *we denote the extension of the renaming* $\imath$ *to the label* μ*);*

2. *if* $\mathsf{S} \xrightarrow{\rho}_{\ell,\ell'} \mathsf{T}$ *then there exists* T' *such that* $\mathsf{S}' \xrightarrow{\rho}_{\ell'',\ell'''} \mathsf{T}'$ *and* $\mathsf{T} \equiv \mathsf{T}'$.

Proof

1. The proof is a straightforward induction on the derivation tree of $\mathsf{S} \xrightarrow{\mu}_{l} \mathsf{T}$.

2. The result is a direct consequence of the first item and of the (communication) rule.

□

Now that the basic transition relation is defined, we can derive the collective transition relation according to the definition 2.1.2. It is illustrated in the following example.

Example 3.1.4 *As we have seen above the solution* $A[e^1](bond), A[e^1](bond), A[e^1](bond), B[e^{-1}](bond)$ *has three outgoing transitions labelled* $\xrightarrow{\rho}_{1,4}$, $\xrightarrow{\rho}_{2,4}$ *and* $\xrightarrow{\rho}_{3,4}$ *to structurally congruent states. Therefore we obtain an unique collective transition:*

$$A[e^1](bond), A[e^1](bond), A[e^1](bond), B[e^{-1}](bond) \overset{300}{\longmapsto} A[e^0](bond^x), A[e^1](bond), A[e^1](bond), B[e^0](bond^x)$$

Finally the downgrading of a nanoκ collective transition relation can be performed according to the definition 2.2.3.

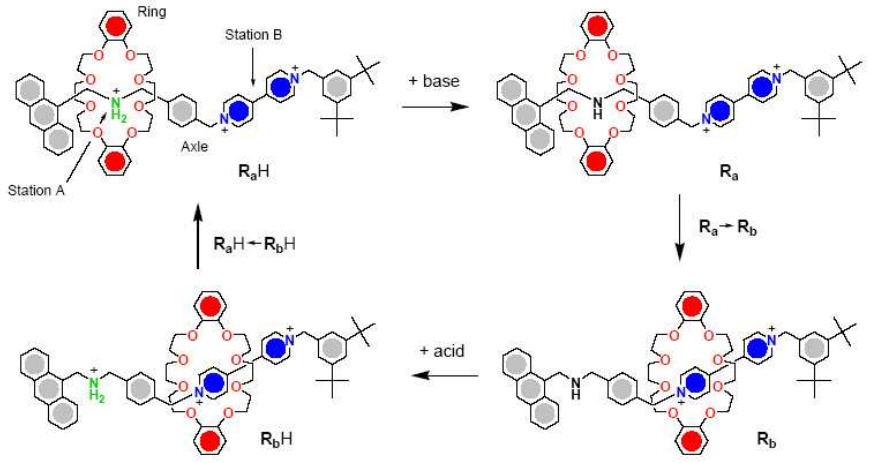


Figure 3.1: Schematic representation of the shuttling processes of the molecular ring in the examined rotaxane.

3.2 The nanoκ calculus at work: the rotaxane case study

The investigated rotaxane RaH (Figure 3.1) [54, 1] is made of a stoppered axle containing an ammonium (A) and an electron acceptor bipyridinium (B) stations that can establish hydrogen-bonding and charge-transfer interactions, respectively, with the ring component, which is a crown ether with electron donor properties. An anthracene moiety is used as a stopper because its absorption, luminescence, and redox properties are useful to monitor the state of the system. Since the hydrogen bonding interactions between the macrocyclic ring and the ammonium center are much stronger than the charge-transfer interactions of the ring with the bipyridinium unit, the rotaxane exists as only one of the two possible translational isomers, denoted as RaH in Figure 3.1. In solution, addition of a base (e.g., tributylamine) converts the ammonium center into an amine function, giving the transient state Ra that is transformed into the stable state Rb as a consequence of the displacement of the macrocycle onto the B station. The process can be reversed by addition of acid (e.g., trifluoroacetic acid) and the initial state is restored, passing through the transient state denoted as RbH. Nuclear magnetic resonance, absorption and luminescence spectroscopic experiments, together with electrochemical measurements, indicate that the acid-base controlled switching, which is fully reversible and relatively fast, exhibits a clear-cut on-off behaviour [1].

The Rotaxane RaH is particularly appropriate to test the modeling approach of the nanoκ calculus because it is one of the very few cases wherein not only the thermodynamic properties, but also the dynamic behaviour of the system

have been experimentally characterized in detail. Specifically, the macrocycle's shuttling process between the ammonium/amine and bipyridinium stations in this rotaxane, driven by the successive addition of base and acid, have been investigated in solution [36]. The rate constants for the "forward" (Ra→Rb) and "backward" (RbH→RaH) shuttling motions (vertical processes in Figure 3.1) of the molecular ring, which occur, respectively, upon deprotonation and reprotonation (that is upon loss or gain of a proton respectively) of the ammonium/amine recognition site on the axle (horizontal processes in Figure 3.1), were found to be $0.72s^{-1}$ and $40s^{-1}$ at $293°K$, respectively.

3.2.1 Modeling the rotaxane RaH in the nanoκ calculus.

The nanoκ calculus molecules. Figure 3.2 illustrates the nanoκ calculus modeling of the rotaxane RaH. We use four species:

- *Nh* models the ammonium/amine station of the rotaxane: it has one field h and two sites *ring* and *axle*;
- *Axle* models the spacer between the two stations: it has two fields h and s and three sites *nh*, *bipy*, and *ring*;
- *Bipy* models the bipyridinium station: it has one field h and two sites *ring* and *axle*;
- *Ring* models the crown ether ring: it has no field and one site *link*;
- *AcidBase* models the acid-base couple used to trigger the motion of the rotaxane: it has one field h and no site.

The pairs of sites *axle* of *Nh* and *nh* of *Axle*, and *axle* of *Bipy* and *bipy* of *Axle* are always linked in our modeling. They model the covalent bonds maintaining the structural integrity of the axle. Exactly one site *ring* of *Nh*, *Bipy*, and *Axle* is linked at a given moment at *link* of *Ring*. The first two cases respectively model the "stable" RaH and Rb states of Figure 3.1 in which the ring is steadily located around the *Nh* or the *Bipy* molecules, respectively. The last case models the "unstable" states; these are the Ra and RbH states of Figure 3.1 in which the ring is not steadily located.

Ammonium and amine functions have different chemical nature but can be seen as protonated and deprotonated versions of the same species. Thus we model both by the same nanoκ calculus species *Nh*. Its field h is used to record the presence or absence of a proton on *Nh*: its value is 1 if it is protonated, and 0 otherwise. We also need to distinguish between the two transient states where the *Ring* is on the *Axle*: does it come from the *Nh* station of from the *Bipy* one? In order to store this information we use the field s: its value is 1 if the *Ring* comes from the *Nh* station and 0 otherwise.

As *Ring*'s movements are triggered by protonations and deprotonations due to acid-base reactions, we also need to have acid and base molecules in our modeling. We choose to model an acid-base couple with only one species since

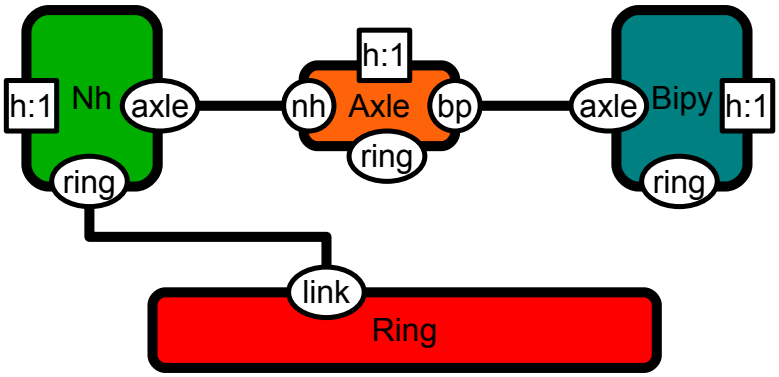


Figure 3.2: Initial state of the Rotaxane RaH in nanoκ calculus.

an acid and a base of the same couple only differ by a proton. We consider the species *AcidBase* with no site and one field h having value 1 in case the acid/base molecule holds the proton to be exchanged, 0 otherwise (for instance $AcidBase[h^1]$ and $AcidBase[h^0]$ are respectively an acid molecule ready to give a proton and a base molecule ready to receive a proton). If a different acid-base were to be considered it would be modeled similarly by a species $AcidBase_2$ with one field h and no site.

The initial state for rotaxane RaH is thus modeled by the term:

$$Nh[h^1](axle^s + ring^x) \ , \quad Axle[h^1 + s^1](nh^s + bipy^r + ring) \ ,$$
$$Bipy[h^1](axle^r + ring) \ , \quad Ring(link^x)$$

graphically depicted in Figure 3.2.

Note that the *Nh* is initially protonated (and this information is present also in the *Axle* and the *Bipy*), the *Axle* is bond to the *Nh* and the *Bipy*, and the *Ring* is bond to the *Nh*.

The nanoκ calculus reactions. We now present the reactions used in our modeling. Reactions 1, 2, 9 and 10 are presented with a double arrow (these are *reversible* reactions). Formally they correspond to two nanoκ calculus reactions, one achieved reading the reaction from left to right considering the rate over the arrow, and the ones achieved reading it from right to left considering the rate below. In this section we do not consider numerical values of rates, this is detailed in part 3.2.2.

A base can get the proton of a protonated *Nh*, and a *Nh* can get a proton from an acid. These acid-base reactions are reversible. Reactions 1 and 2 model

this phenomenon. The systems corresponding to the left-hand side and right-hand side coexist, even if one can be more predominant according to the ratio $nh\_base/base\_nh$.

$$Nh[h^1], AcidBase[h^0] \underset{base\_nh}{\overset{nh\_base}{\rightleftarrows}} Nh[h^0], AcidBase[h^1] \quad (1)$$

The protonation state of the molecule *Nh* needs to be known by *Bipy* because it affects its interaction with *Ring*. Reactions 3 and 4 achieve this by passing information from *Nh* to *Bipy* through *Axle*. These updates are instantaneous because they represent an immediate consequence of the protonation or deprotonation of the Nh station.

if $(\alpha \neq \beta)$
$Nh[h^\alpha](axle^s), Axle[h^\beta](nh^s) \overset{\infty}{\rightarrow} Nh[h^\alpha](axle^s), Axle[h^\alpha](nh^s) \qquad (3_{\alpha,\beta})$
and:
$Axle[h^\alpha](bipy^r), Bipy[h^\beta](axle^r) \overset{\infty}{\rightarrow} Axle[h^\alpha](bipy^r), Bipy[h^\alpha](axle^r) \qquad (4_{\alpha,\beta})$

The above rule correspond actually to many rules, one for each possible value of α and β. We gather them in two rules for the sake of the clarity. We achieve the modeling of *Ring* movements in two steps. Firstly the instantaneous reactions to deprotonation/reprotonation (reactions 5–8), and secondly the actual *Ring* shuttling (reactions 9 and 10). The reactions (5) and (6) are used to enter in "unstable" states when the *Nh* is deprotonated while the *Ring* is around the *Nh* (reaction (5)), or protonated while the *Ring* is around the *Bipy* (reaction (6)). On the other hand, the reactions (7) and (8) are used to re-enter in a "stable" state in the case the *Nh* returns to its previous (de)protonated state before the *Ring* actually binds to its new station. All these events are immediate consequences of deprotonation or reprotonation of *Nh*; for this reason, they have infinite rates. When a field contains a $*$, it means that there is a rule for each possible value of the field.

$$Nh[h^0](axle^s + ring^x), Axle[s^*](nh^s + ring) \overset{\infty}{\rightarrow} Nh[h^0](axle^s + ring), Axle[s^1](nh^s + ring^x) \quad (5)$$
$$Bipy[h^1](axle^r + ring^x), Axle[s^*](biax^r + ring) \overset{\infty}{\rightarrow} Bipy[h^1](axle^r + ring), Axle[s^0](biax^r + ring^x) \quad (6)$$
$$Axle[s^1](nh^s + ring^x), Nh[h^1](axle^s + ring) \overset{\infty}{\rightarrow} Axle[s^1](nh^s + ring), Nh[h^1](axle^s + ring^x) \quad (7)$$
$$Axle[s^0](nh^s + ring^x), Bipy[h^0](axle^s + ring) \overset{\infty}{\rightarrow} Axle[s^0](nh^s + ring), Bipy[h^0](axle^s + ring^x) \quad (8)$$

We now complete our modeling with reactions 9 and 10 representing the completion of the *Ring* movement. These reactions are reversible because the *Ring*

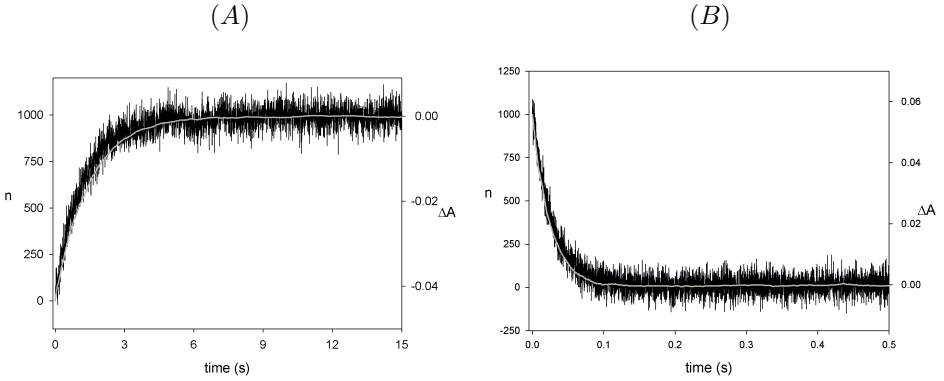


Figure 3.3: Comparing the simulations *in silico* with the experiments *in vitro*. Grey traces: number of *Ring*s located around *Bipy*s during the "forward" Ra→Rb (part A) and the "backward" RbH→RaH (part B). Black traces: UV absorbance changes observed upon the occurrence of the same respective shuttling processes.

is susceptible to leave its "stable" station due to the Brownian motion.

$$Axle[s^*](bipy^r + ring^x)\,,Bipy[h^0](axle^r + ring) \underset{unlink\_bipy}{\overset{link\_bipy}{\rightleftarrows}} Axle[s^0](bipy^r + ring)\,,Bipy[h^0](axle^r + ring^x) \tag{9}$$

$$Axle[s^*](nh^s + ring^x)\,,Nh[h^1](axle^s + ring) \underset{unlink\_nh}{\overset{link\_nh}{\rightleftarrows}} Axle[s^1](nh^s + ring), Nh[h^1](axle^s + ring^x) \tag{10}$$

3.2.2 Simulation results.

The above modeling of rotaxane RaH in `nano`κ calculus yields an IMC system that is strictly markovian and so it can be downgraded to an equivalent CTMC. Therefore we obtain a CTMC system that we use to simulate *in silico* the behaviour of the rotaxane RaH. The simulations are performed using the SPiM tool [19] using the encoding from the `nano`κ calculus to the stochastic π-calculus of the Chapter 4. We did not use the κ-factory because at the time we performed the simulations it were not able to handle infinite rates.

As previously discussed the rates for the ring movements are respectively $link\_bipy = 0.72s^{-1}$ and $link\_nh = 40s^{-1}$. On the basis of the estimated equilibrium constants, the rates for the reverse reactions are quantified two orders of magnitude smaller, i.e. $unlink\_bipy = 0.0072s^{-1}$ and $unlink\_nh = 0.4s^{-1}$.

The aim of the first two simulations depicted in Figure 3.3 is to check whether the experimentation *in silico* can reproduce the results observed in *in vitro* [36]. The techniques used for the *in vitro* experimentation did not make it possible to

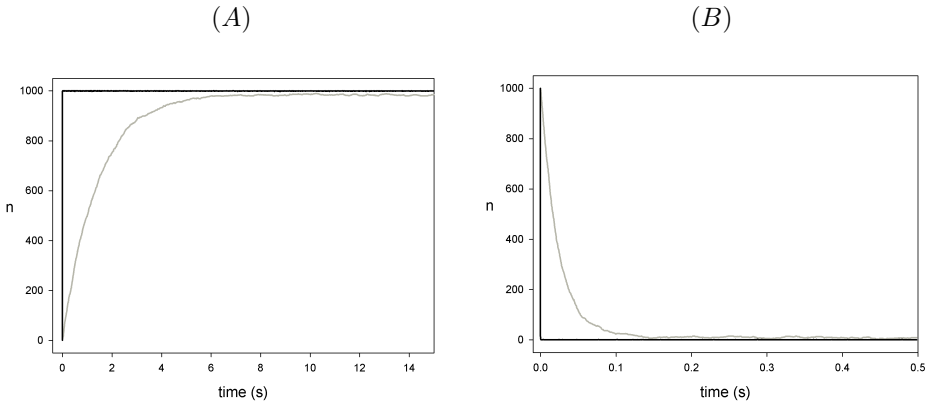


Figure 3.4: Number of *Ring*s located around *Bipy*s (grey trace) and number of deprotonated rotaxanes (black trace) during the "forward" shuttling in the presence of base molecules (part A) and the "backward" shuttling in the presence of acid molecules (part B) at concentration $10^{-4}M$.

observe and quantify the deprotonation/reprotonation rates (this is not surprising as these are very fast acid-base reactions). Thus, in the simulation we have considered instantaneous deprotonation/reprotonation, i.e. either $nh\_base = \infty$ and $base\_nh = 0$ for protonation, or $nh\_base = 0$ and $base\_nh = \infty$ for deprotonation. In both simulations, we have considered 1000 rotaxanes: in the first one we have simulated deprotonation and "forward" (Ra→Rb) shuttling, in the second one reprotonation and "backward" (RbH→RaH) shuttling. In the first simulation the shuttling phase is completed in around 6 seconds, while in the second one in 0.1 seconds; this is a consequence of the different rates of the two directions of shuttling. Very remarkably, simulated data are in striking agreement with the experimental results.

After these initial encouraging results, we have decided to use the *in silico* simulation techniques to provide a comprehensive view of the overall reactions depicted in Figure 3.1, simulating also the deprotonation/reprotonation phases not observed in the *in silico* experimentation. More precisely, the aim of this second group of simulations was to either validate or invalidate the assumption according to which deprotonation/reprotonation can be considered "instantaneous" with respect to the shuttling time. To this aim, we have simulated deprotonation/reprotonation under two different concentrations of rotaxanes. In fact, this is a bimolecular reaction whose rate is influenced by the concentration of the reactants. For instance, at a concentration close to those considered in [36], e.g. $10^{-4}M$, assuming 1000 instances of rotaxane and base/acid, a plausible rate for deprotonation/reprotonation is $2 \times 10^{3}s^{-1}$ (with reverse reaction rate of the order of $2 \times 10^{-4}s^{-1}$) while at the concentration $10^{-8}M$ it is $0.2s^{-1}$ (with reverse reaction on the order of $0.2 \times 10^{-7}s^{-1}$).

We have performed the two simulations, namely deprotonation with subse-

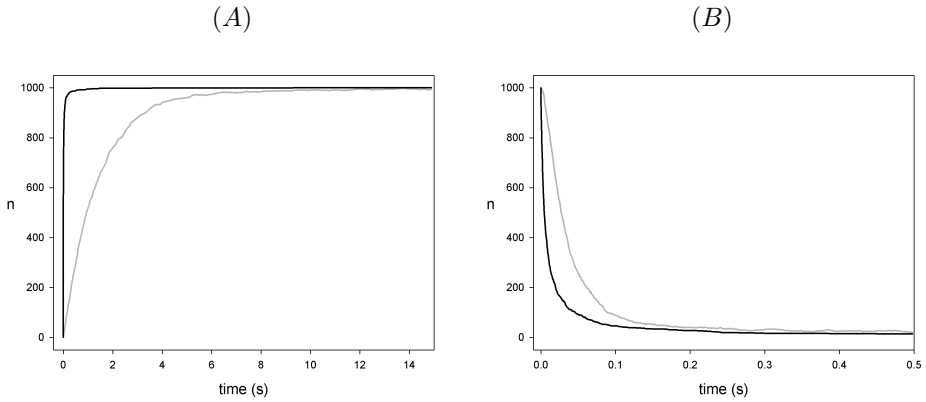


Figure 3.5: Number of *Ring*s located around *Bipy*s (grey trace) and number of deprotonated rotaxanes (black trace) during the "forward" shuttling in the presence of base molecules (part A) and the "backward" shuttling in the presence of acid molecules (part B) at concentration $10^{-8}M$.

quent "forward" shuttling and reprotonation with subsequent "backward" shuttling, considering the two different concentrations.

The results at concentration $10^{-4}M$ are reported in Figure 3.4; they essentially confirm the validity of the "instantaneous" deprotonation/reprotonation assumption at this concentration level. We report in Figure 3.5 the results for concentration $10^{-8}M$; in this case the rings start moving before the deprotonation/reprotonation phase is over. This proves that in the rotaxane RaH the stimulus and the subsequent shuttling could interplay.

In the light of this observation, we have decided to investigate some additional scenarios not yet considered in the *in vitro* experimentations. In particular, we have decided to analyze the interplay between shuttling and a stimulus given by *weaker* acid/base molecules, that is, for which the ratio between the deprotonation/reprotonation rate and the reverse rate is smaller. In fact, the ratio considered in the previously discussed simulations is on the order of 10^7; a smaller reasonable ratio could be on the order of 10^3. Considering this new ratio, assuming 1000 instances of rotaxane and base/acid, at the concentration $10^{-4}M$ the new rates for deprotonation/reprotonation is $2\times10^3 s^{-1}$ with reverse reaction rate on the order of $2s^{-1}$, while at the concentration $10^{-8}M$ it is $0.2s^{-1}$ with reverse reaction on the order of $0.2 \times 10^{-3}s^{-1}$. Using these new rates, we have simulated the "forward" and "backward" shuttling at both concentrations, $10^{-4}M$ in Figure 3.6 and $10^{-8}M$ in Figure 3.7.

Interestingly, we found out that the "forward" shuttling is no longer guaranteed. In fact, only in some of the deprotonated rotaxanes the *Ring* actually moves around the *Bipy*. In other terms, the efficiency of the rotaxane is no longer close to 100% (as was the case in the *in vitro* experimentations and in the other *in silico* simulations) but it is around 35% for concentration $10^{-4}M$,

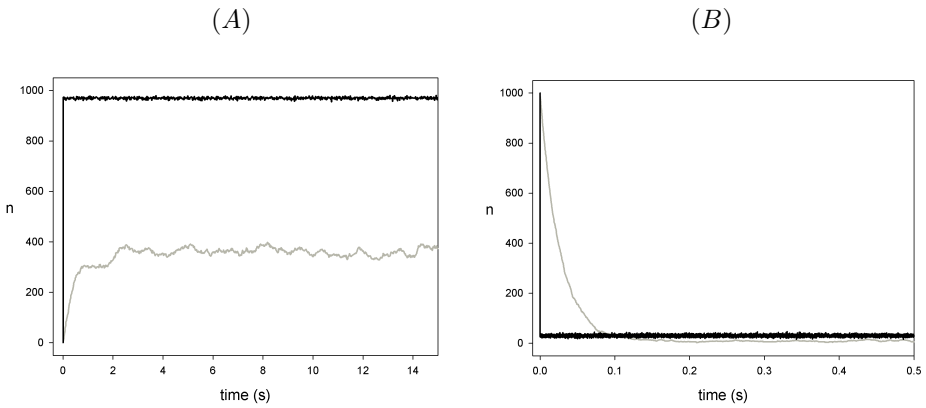


Figure 3.6: Number of *Ring*s located around *Bipy*s (grey trace) and number of deprotonated rotaxanes (black trace) during the "forward" shuttling in the presence of weak base molecules (part A) and the "backward" shuttling in the presence of weak acid molecules (part B) at concentration $10^{-4}M$.

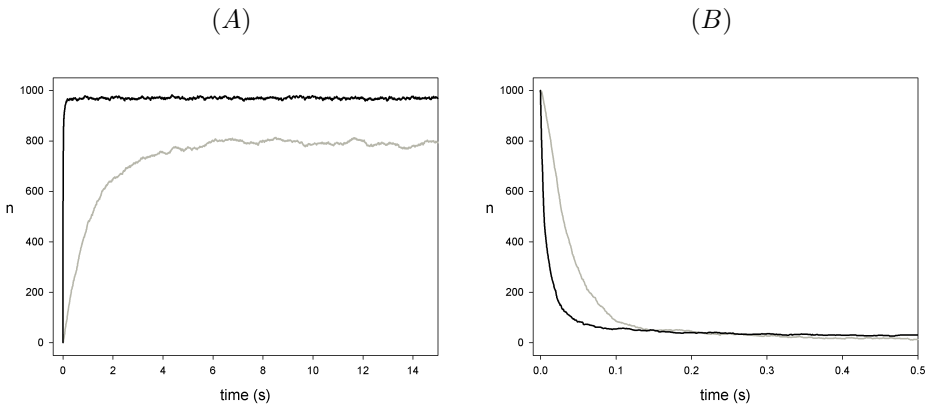


Figure 3.7: Number of *Ring*s located around *Bipy*s (grey trace) and number of deprotonated rotaxanes (black trace) during the "forward" shuttling in the presence of weak base molecules (part A) and the "backward" shuttling in the presence of weak acid molecules (part B) at concentration $10^{-8}M$.

or 75% for concentration $10^{-8}M$. After an analysis of this initially unexpected results, we can conclude that the inefficiency of the rotaxane is justified by the fact that the reverse reaction of deprotonation (i.e. re-protonation) can activate a chain of reactions that allows an already deprotonated rotaxane, with the *Ring* around the *Bipy*, to return in the initial state (protonated with the *Ring* around the *Nh*). This chain of reactions, under these particular circumstances, plays an important role in the equilibrium between the number of deprotonated rotaxanes with the *Ring* around the *Nh* and the number of deprotonated rotaxanes with the *Ring* around the *Bipy*.

3.3 Conclusion.

We have introduced `nano`κ, a calculus designed on purpose for the modeling of nano-devices. The calculus is equipped with a stochastic semantics (defined in terms of a CTMC) that can be used to simulate the evolution of the behaviour of nano-devices using stochastic simulation techniques such as, for instance, the Gillespie algorithm [37]. We have applied the `nano`κ calculus to the modeling and simulation of the RaH rotaxane [54, 1], a nano-device that attracted a lot of attention inside the nano science and technology community, because it proved very useful for building more complex nano-devices [48, 46, 2]. We have used the `nano`κ calculus model of the RaH rotaxane to simulate its behaviour under conditions that were not yet considered in the *in vitro* experimentations. We found out that under particular circumstances the nano-device is not as efficient as expected. In particular, even if almost all the rotaxanes in a solution are stimulated, only some of them change their internal structure according to the stimulus.

As future work, we intend to use the `nano`κ calculus to model and simulate also more complex nano-devices, such as the nano-elevator [2]. As we detailed in Chapter 1, nano elevators are composed of a platform and of three rotaxanes that, once appropriately stimulated, move the platform up or down. We expect to reuse the modeling of the rotaxane presented in this paper. In fact, one of the most important peculiarities of the `nano`κ calculus is that it supports compositional modeling: the reactions describing the behaviour of the molecules that are part of a nano-device, are still valid reactions also when the nano-device is itself considered as a part of a more complex system.

We have already discussed in the Introduction the origins of the `nano`κ calculus, and its strong relationship with the κ-calculus[26]. Here we simply recall that the `nano`κ calculus can be seen as a member of the κ-family. The κ-calculus benefits from efficient techniques of simulation and analysis [25, 24, 47, 27]. In contrast to our `nano`κ calculus this formalism allows reactions involving an arbitrary number of molecules, but there are no exchanges rules, edges can only be created or destroyed, not moved. These differences are explained by our field of application. Dealing with the behaviour of nano-complexes, the relevant reaction we met involve barely more than two molecules, but edges are often exchanged and moved between molecules.

3.4 Related works.

The nanoκ calculus has been influenced also by Cardelli's language of stochastic interacting processes [14, 17] that has been put in correspondence with Ordinary Differential Equations. The stochastic semantics of the nanoκ calculus, indeed, has been given following these lines.

Another process calculus for the modeling of biochemical systems is Bio-PEPA [22, 7]. Differently from the Cardelli's approach, there is no one-to-one correspondence between processes and molecules, but one process is used to represent the concentration of one species. In Bio-PEPA the rates are associated to the actions by means of "functional rates": these are functions that are evaluated at the moment of the reduction of the systems. The idea of functional rates is particularly useful when different kinetic laws are considered in the same unifying framework. The possibility of considering different kinetic laws is also proposed in BIOCHAM [13], a programming environment for modeling biochemical systems, making simulations and querying the model in temporal logic. Our approach is different from both Bio-PEPA and BIOCHAM because we follow the Cardelli's one-to-one correspondence between molecules and processes. In fact, we have found this approach appropriate for a compositional model of discrete state systems (in which we count the number of molecules instead of considering their concentrations).

The beta-binders [63] which evolved recently into the BlenX language [30] are another formalism that can represent complexing molecules. It is based on a π-calculus where the usual communication discipline is relaxed to better represent the complementarity of molecular binding sites. It is achieved by means of a wrapping operator associating an interface to a group of π-processes.

Finally, in the *calculus of looping sequences* [6, 4] a different paradigm is taken. Molecules are represented simply by a name rather than by a π-process and they can be assembled in sequences. Closed chains of molecules are used to represent membranes, while dynamics is governed by rewriting rules on names.

Chapter 4

From biochemistry to stochastic processes.

4.1 Introduction.

As we detailed in the Chapter 1, two main approaches emerged for the formal representation of biochemical systems: the *rule-based* approach [13, 25, 5, 21, 12] and the *process-oriented* approaches [15, 43, 16]. The former is inspired by the traditional chemistry kinetics while the latter is inspired by process calculi. In brief, the advantage of the rule-based approach is that it provides a biochemical-like syntax as well as simple and easily reusable modelings. The advantage of the process-oriented approach is that it comes with all the theory and tools that have been developed for process calculi [39, 15, 43, 74, 45].

In this chapter we bridge the gap between the two approaches by implementing the `nano`κ calculus presented in the previous Chapter, which is a rule-based formalism, into a process-oriented formalism: the `nano`π-calculus, a subset of the stochastic π-calculus. The `nano`π-calculus is a nice target language for several reasons. First it is a subcalculus of the π-calculus and so we are optimistic that it can benefit of much of the known theory and available tools that have been developed for the π-calculus (see for instance [39, 74, 45]). Secondly it corresponds to the core of the SPiM simulator: a simulator for the stochastic π-calculus [19, 15]. And finally it is a natural and conservative extension of the *Chemical Ground Form* calculus, for which Cardelli established a close correspondence with systems of chemical equations [14] (see the related works paragraph for some more details).

Despite the simplicity of the `nano`κ reactions, their implementation in `nano`π-calculus is complex if the stochastic semantics must be preserved. We present and discuss the problems through a number of examples. We start with a simple yet defective proposition of encoding that we upgrade three times until we reach a correct encoding.

An implementation of `nano`κ into `nano`π should project the behavior of each molecule out of the set of reactions and collect them into a process definition. For example, consider a set of reactions ρ_1, ρ_2, ρ_3, ... If ρ_1 is:

$$\rho_1 \qquad A[s^0](a), B[t^1](b) \overset{\lambda_1}{\rightarrow} A[s^1](a^x), B[t^0](b^x)$$

then the `nano`π process $\widehat{\texttt{A}}$ of the molecule A in the above example is:

$$\begin{aligned} \widehat{\texttt{A}}(s,a) \;=\; & \texttt{behavior-of A in } \rho_1 \\ & + \texttt{behavior-of A in } \rho_2 \\ & + \texttt{behavior-of A in } \rho_3 \\ & + \ldots \end{aligned}$$

That is, a molecule is implemented by a parametric process definition $\widehat{A}$, whose parameters s and a encode the values of the corresponding fields and sites. The "`behavior-of A in` ρ_1" might be defined as

$$[a = \varepsilon, s = 0]\overline{\rho_1}\, x.\widehat{\texttt{A}}(1, x)$$

where

- $[a = \varepsilon, s = 0]$ means that such a behavior may be triggered provided the site a is unbound (has value ε) and the field s is equal 0;

- in this case the name of the reaction ρ_1 is used as a channel on which a fresh name (modeling the bond created) is output. The channel ρ_1 is expected to be declared in the context and to have rate λ_1. We also expect that in the definition $\widehat{\mathtt{B}}$ the `behavior-of A in` ρ_1 performs a corresponding input on ρ_1 when the field t is 1 and the site b is unbound;

- then $\widehat{\mathtt{A}}$ will continue as the process $\widehat{\mathtt{A}}(1, x)$, since the values of the field s and site a after the reaction are respectively 0 and x.

However the analogy *reaction-names as channels* cannot be pushed forward to destruction. In fact, supposing for instance that ρ_2 is the following destruction:

$$\rho_2 \qquad A[s^1](a^x), B[t^0](b^x) \stackrel{\lambda_2}{\to} A[s^0](a), B[t^1](b)$$

if the "`behavior-of A in` ρ_2" was modeled as

$$[a = \neg\varepsilon, s = 1]\ \overline{\rho_2}\,().\widehat{\mathtt{A}}(0, \varepsilon)$$

then $\widehat{\mathtt{A}}$ might interact with the wrong $\widehat{\mathtt{B}}$ in the `nano`π implementation ($a = \neg\epsilon$ means that a is bond). This difficulty may be circumvented by using the names encoding bonds – that are shared exactly by the two connected molecules – in order to send the disconnection signal. So the "`behavior-of A in` ρ_2" becomes

$$[a = \neg\varepsilon, s = 1]\ \overline{a}\,().\widehat{\mathtt{A}}(0, \varepsilon)$$

and we are assuming that $\widehat{\mathtt{B}}$ will input on a and that the rate of this name is λ_2.

Unfortunately, this solution is also inadequate, because some other reaction may address the site a of `A`. If for instance the reaction ρ_3 is:

$$\rho_3 \qquad A[s^1](a^x), B[s^1](b^x) \stackrel{\lambda_3}{\to} A[s^0](a^x), B[s^0](b^x)$$

with $\lambda_3 \neq \lambda_2$. Because of this inequality, it is not possible to use the channel x anymore. One should rather use a channel for every reaction addressing the bond. In the above example:

$$\begin{aligned}\widehat{\mathtt{A}}(s, a.\rho_2, a.\rho_3) \;=\; & [a.\rho_2 = \varepsilon, a.\rho_3 = \varepsilon, s = 1]\ \overline{\rho_1}\,(x_2 : \lambda_2, x_3 : \lambda_3).\ \widehat{\mathtt{A}}(0, x_2, x_3) \\ & + [a.\rho_2 = \neg\varepsilon, s = 0]\ \overline{a.\rho_2}\,().\ \widehat{\mathtt{A}}(1, \varepsilon, \varepsilon) \\ & + [a.\rho_3 = \neg\varepsilon, s = 1]\ \overline{a.\rho_3}\,().\ \widehat{\mathtt{A}}(0, a.\rho_2, a.\rho_3) \\ & + \cdots\end{aligned}$$

where the parameter $a.\rho_2$ and $a.\rho_3$ contain the channels corresponding to the reactions ρ_2 and ρ_3 addressing the site a. Moreover these parameters are instantiated in the branch corresponding to the rule ρ_1 because this rule creates the bond on a.

Unfortunately, a last problem renders this solution defective. Consider the rules:

$$\rho_4 \qquad A(a^x), C(c) \overset{\lambda_4}{\to} A(a), C(c^x)$$
$$\rho_5 \qquad B(b^x), C(c^x) \overset{\lambda_5}{\to} B(b), C(c)$$

In this case the following scenario is possible: thanks to the rule ρ_1 a bond is created between A and B, then thanks to ρ_4 its A-extremity is flipped toward C and then thanks to ρ_5 the bond is destroyed. As we have argued before, the encoding of B and C need to share a private channel dedicated to the modeling of the rule ρ_5. However before the firing of ρ_4 `B` and `C` were not linked, so they should not share such a channel. So the sharing of such a channel has to result from the firing of ρ_4. Our solution is to model the creation of a bond by the creation of a channel for every reaction in the `nano`κ system, and to exchange or destroy the whole tuple of these channels whenever the bond is exchanged or destroyed respectively.

Actually in an accurate solution, a bond in `nano`κ should be represented in `nano`π by a tuple whose length is the number of reactions that address that bond directly or indirectly through sequences of exchanges: it corresponds to *the computation of all the possible future events* and the creation of a dedicated channel for each such event. But for the sake of the simplicity our solution over-approximates the "precise" solution, by representing bonds with tuples whose length is the size of the set of reactions – the *gangs*, in our terminology.

The encoding of `nano`κ into `nano`π defined in this Chapter, written $[\![\cdot]\!]$, is such that $\mathsf{S} \overset{\lambda}{\longmapsto}_{\mathtt{n}\kappa} \mathsf{T}$ if and only if $[\![\mathsf{S}]\!] \overset{\lambda}{\longmapsto}_{\mathtt{n}\pi} [\![\mathsf{T}]\!]$ (the arrow is subscribed to ease the reading). It follows that S and $[\![\mathsf{S}]\!]$ are *strongly stochastic bisimilar* [8]. Actually our correctness property is even stronger: it corresponds to the case of a strong stochastic bisimulation where the bisimulation relation is bijective. This could be seen as some sort of homomorphism.

This is different from usual implementations that almost never preserve the granularity of transitions, that is they encode one transition into several. Actually in the stochastic settings preserving the granularity is quite natural for an encoding. This is because the stochastic rate attached to a transition is the parameter of an exponential law governing the waiting time before the transition could be fired. Unfortunately the sum of two exponential law is not an exponential law. So it seems unlikely that an encoding which does not preserve the granularity could preserve the distribution of the sojourn time.

Infinite rates offer, however, an alternative: the time needed to perform a sequence of transitions in which only one has a finite rate is the same as the time to perform one transition with the same rate. But the non-determinism introduced by the infinite rates make such an encoding hard to achieve.

The Chapter is organized as follows. In Section 4.2, we introduce the syntax and semantics of the `nano`π-calculus. In Section 4.3, we define our encoding between the `nano`κ calculus and the `nano`π-calculus. In Section 4.4, we illustrate

our encoding on an example. In Section 4.5, we collect the correctness theorems of our encoding. The Chapter is closed by a conclusion in Section 4.6 and a discussion on related works in Section 4.7.

4.2 The nanoπ-calculus.

Syntax of the nanoπ-calculus.

Definition 4.2.1 (Processes) *The* nanoπ*-calculus uses two sets of identifiers:* channel names*, which are totally ordered, include the natural integers*[1] *and are ranged over by* x*,* y*,* u*,* $\cdots$*, and* agents*, ranged over by* A*,* B*,* $\cdots$*. Channel names have a rate that is a positive real number or* ∞*. This rate may be explicitly declared in the process or globally defined (for free names). The following syntactic categories are used in the* nanoπ*-calculus:*

$$\begin{array}{rcll} M & ::= & [u=v] \quad | \quad M\,M & \textit{matches} \\ \alpha & ::= & x\,(\widetilde{u}) \quad | \quad \overline{x}\,\widetilde{u} \quad | \quad \overline{x}(\widetilde{u}:\widetilde{\lambda}) & \textit{actions} \\ P & ::= & \mathbf{0} \quad | \quad A(\widetilde{u})|P & \textit{terms} \end{array}$$

Moreover agent declarations have the form:

$$A(\widetilde{x}) \triangleq \sum_{i\in I} M_i\alpha_i.P_i$$

and a process is a term $(\widetilde{x}:\widetilde{\lambda})\ P$ *where* $\widetilde{\lambda}$ *are rates.*

Matches are sequences of equalities between values. An action is either an *input* $x\,(\widetilde{u})$ on the channel x of a tuple of names $\widetilde{u}$, or an *output* $\overline{x}\,\widetilde{u}$ on x of a tuple $\widetilde{u}$, or an *bond output* $\overline{x}\,(\widetilde{u}:\widetilde{\lambda})$ on x of a tuple of new names $\widetilde{u}$ with rates $\widetilde{\lambda}$. Terms can be the inert $\mathbf{0}$ or a parallel composition of agent invocations. The parallel operator $|$ is assumed to be associative.

The term $\widetilde{x}:\widetilde{\lambda}$ has to be considered a set with the constraint that every two different elements have different names. Processes are ranged over by P, Q, $\cdots$.

Scope restrictions bind names, that is in $(x:\lambda)\ P$ the x free in P is bond by $x:\lambda$. Likewise, input $x\,(\widetilde{u}).P$ and bond output $\overline{x}\,(\widetilde{u}:\widetilde{\lambda}).P$ bind $\widetilde{u}$ with scope P. The agent definition $A(\widetilde{u}) \triangleq \sum_{i\in I} M_i\alpha_i.P_i$ binds $\widetilde{u}$ with scope equal to the right hand side of the definition. We write $\mathtt{bn}(T)$ and $\mathtt{bn}(\alpha)$ for the set of the bonded names of a process T or of an action α. The bonded names of an action α is $\mathtt{bn}(\alpha)$ defined by $\widetilde{u}$ if α is either $x\,(\widetilde{u})$ or $\overline{x}\,(\widetilde{u}:\widetilde{\lambda})$, and by $\emptyset$ if $\alpha = \overline{x}\,\widetilde{u}$. Names that are not bound are called *free* and we write $\mathtt{fn}(T)$ for the set of such names in T.

We assume that all terms meet the following well formed properties:

- in $(\widetilde{x}:\widetilde{\lambda})P$, $\widetilde{x} \subseteq \mathtt{fn}(P)$ (there is no garbage);
- bond names in agent definitions never clash with free names (this allows us to avoid alpha-conversions).

[1] We ask that channels can be natural integers so that they can represent the values of the fields of molecules.

Notation. Whenever a match has the form $[u = u]$, or a sum has only one branch we omit to write them explicitly. For instance $A(\widetilde{x}) \triangleq \sum_{i\in\{1\}} [\widetilde{x}_i = \widetilde{x}_i]\alpha.P$ is written $A(\widetilde{x}) \triangleq \alpha.P$.

Given any nanoπ term P, and any tuples of names $\widetilde{u}$ and $\widetilde{v}$, we define $P\{\widetilde{v}/\widetilde{u}\}$ as the term P where the names of the tuple $\widetilde{u}$ have been substituted by the names of the tuple $\widetilde{v}$. We can now present the structural congruence for the nanoπ-calculus:

Definition 4.2.2 *The relation $\equiv$ is the smallest equivalence on the terms of the* nanoπ*-calculus such that (recall that the solutions are already quotiented by the associativity of $|$):*

- $(\widetilde{x} : \widetilde{\lambda})(P|Q) \equiv (\widetilde{x} : \widetilde{\lambda})(Q|P)$*: i.e. the parallel operator is commutative,*
- $P \equiv Q$ *if Q is an α-renaming of P, that is given a tuple of bound names $\widetilde{x}$ of P and a tuple of names $\widetilde{z}$ which do not appear in P, $Q = P\{\widetilde{z}/\widetilde{x}\}$,*
- $0|P \equiv P$*,*
- *for any permutation σ of the set I, $\sum_{i\in I} \alpha_i.P_i \equiv \sum_{i\in I} \alpha_{\sigma(i)}.P_{\sigma(i)}$: that is the choice is commutative.*

Stochastic semantics of the nanoπ-calculus. The basic transition relation of the nanoπ-calculus requires a few definitions:

- M *is true* if M is a sequence of $[x = x]$.
- *length*$(A_1(\widetilde{u_1}) \mid \cdots \mid A_n(\widetilde{u_n}))$ returns n.
- Given two name creation $(\widetilde{x} : \widetilde{\lambda})$ and $(\widetilde{y} : \widetilde{\lambda'})$, the name creation $(\widetilde{x} : \widetilde{\lambda} + \widetilde{y} : \widetilde{\lambda'})$ is defined as the sequence $z_1 : \lambda_1, \cdots, z_n : \lambda_n$ where $z_1, \cdots, z_n$ are pairwise different names, $\{z_1, \cdots, z_n\} = \widetilde{x} \cup \widetilde{y}$, and $z_i : \lambda_i$ if either $(z_i : \lambda_i) \in \widetilde{y} : \widetilde{\lambda'}$ or $z_i \notin \widetilde{y}$ and $z_i : \lambda_i \in \widetilde{x} : \widetilde{\lambda}$.
- $[(\widetilde{x} : \widetilde{\lambda})P]_{\mathtt{GC}} = (\widetilde{z} : \widetilde{\lambda'})P$ such that $(y : \lambda'') \in (\widetilde{z} : \widetilde{\lambda'})$ if $y \in \mathtt{fn}(P)$ and $y : \lambda''$ is in $\widetilde{x} : \widetilde{\lambda}$: that is the operation $[(\widetilde{x} : \widetilde{\lambda})P]_{\mathtt{GC}}$ removes the declaration of the names that are not used P.
- Given a name renaming $\imath$, with an abuse of notation we lift it to a tuple of names or to a process by applying it pointwise.

Definition 4.2.3 *The basic transition relation of the* nanoπ*-calculus, written either $\xrightarrow{\alpha}_{l.i,l'.j}$ or $\xrightarrow{\tau_\lambda}_{l.i,l'.j}$ or $\xrightarrow{\alpha}_{l.i}$, is the least one satisfying the following rules:*

- *(init) let $A(\widetilde{u}) = \sum_{i\in I} M_i\alpha_i.P_i$. If the matches $M_j\{\widetilde{v}/\widetilde{u}\}$ are true, then $A(\widetilde{v}) \xrightarrow{\alpha_j\{\widetilde{v}/\widetilde{u}\}}_{1.j} P_j\{\widetilde{v}/\widetilde{u}\}$;*

- *(lifts) if* $P \xrightarrow{\alpha}_{l.i} P'$ *and* $\mathtt{bn}(\alpha) \cap \mathtt{fn}(Q) = \emptyset$ *and* $l' = length(Q)$*, then both:*
  - $P \mid Q \xrightarrow{\alpha}_{l.i} P' \mid Q$ *and*
  - $Q \mid P \xrightarrow{\alpha}_{l'+l.i} Q \mid P'$*;*
- *(communication) let* $l'' = length(P)$*,* λ *be the rate of* x*, and* $Q \xrightarrow{x\,(\widetilde{u})}_{l'.i'} Q'$*. If* $P \xrightarrow{\overline{x}\,\widetilde{v}}_{l.i} P'$ *then:*
  - $(\widetilde{z} : \widetilde{\lambda'})(P \mid Q) \xrightarrow{\tau_\lambda}_{l.i,l'+l''.i'} [(\widetilde{z} : \widetilde{\lambda'})(P \mid Q\{\widetilde{v}/\widetilde{u}\})]_{GC}$.

  If $P \xrightarrow{\overline{x}\,(\widetilde{v}:\widetilde{\lambda''})}_{l.i} P'$ *then*

  - $(\widetilde{z} : \widetilde{\lambda'})(P|Q) \xrightarrow{\tau_\lambda}_{l.i,l'+l''.i'} [(\widetilde{z} : \widetilde{\lambda'}+\widetilde{\iota(v)} : \widetilde{\lambda''})(P'\{\widetilde{\iota(v)}/\widetilde{v}\}|Q'\{\widetilde{\iota(v)}/\widetilde{u}\})]_{\mathtt{GC}}$

  where ι *is an order-preserving injective renaming that maps* $\widetilde{v}$ *to the least names not belonging to* $\mathrm{name}(P \mid Q) \setminus \{\widetilde{v} \bigcup \widetilde{u}\}$*. Symmetrically when* P *performs an input and* Q *performs an output.*

As for nanoκ, there is always at most one $(\widetilde{z} : \widetilde{\lambda'})P'$ such that $(\widetilde{x} : \widetilde{\lambda})P \xrightarrow{\tau_{\lambda''}}_{l,i,l',i'} (\widetilde{z} : \widetilde{\lambda'})P'$ because created names are the least possible ones and because the renamings are order-preserving.

Now that the basic transition relation is defined, the collective transition relation can be derived according to the definition 2.1.2, and the downgrading can be performed according to the definition 2.2.3.

4.3 Encoding the nanoκ calculus into the nanoπ-calculus.

The definition of the encoding of the nanoκ calculus into the nanoπ-calculus is presented in two steps. The first one defines an internal translation of nanoκ calculus that expands every bond into tuples of bonds. The bonds in the tuple are an over-approximation of the reactions that use the bond. We call these tuples of newly generated names *gangs*. The second step defines a translation from nanoκ (with gangs) to nanoπ.

Before proving the correctness of the encoding we will illustrate it on an example.

4.3.1 Gangs: a dedicated name for every reaction.

In the following we use tuples of the same length n, which is equal to the number of reactions in the nanoκ system we want to encode. These tuples are ordered as follows: $(x_1, \ldots, x_n) \leq (y_1, \ldots, y_n)$ if and only if, for every i, $x_i \leq y_i$. We ask that the set of these tuples is totally ordered: for all tuples $\widetilde{x}$ and $\widetilde{y}$, either $\widetilde{x} \leq \widetilde{y}$ or $\widetilde{y} \leq \widetilde{x}$. Let ε^n be the tuple of n elements ε and let $\jmath$ be a bijective

function that maps ε to ε^n and names to n-tuples of names such that $x \leq y$ if and only if $\jmath(x) \leq \jmath(y)$.

Remarque 5 *Such a set of tuples and such a function $\jmath$ exist because the set of names is totally ordered and countable. Indeed, since the set of names is totally ordered and countable we can write it $(x_k)_{k\in[\![1\ldots\infty[\![}$. Then it suffices to define $\jmath(.)$ in the following way: $\jmath(x_k)$ is the tuple $\{x_{(k-1)\times n+1}, \ldots, x_{k\times n}\}$ and $\jmath(\varepsilon)$ is ε^n.*

We do have that this set of tuples it totally ordered and that the function $\jmath$ is bijective from $\{(x_k)_{k\in[\![1\ldots\infty[\![} \bigcup\{\varepsilon\}\}$ to $\{\jmath(x_k) \,|\, k \in [\![1\ldots\infty[\![\} \bigcup\{\varepsilon^n\}$ and such that $x \leq y$ if and only if $\jmath(x) \leq \jmath(y)$.

Definition 4.3.1 *Let $\mathcal{R} = \{\rho_i : \mathsf{L}_i \overset{\lambda_i}{\rightarrow\!\!\!\!\triangleright} \mathsf{R}_i \mid i \in 1..n\}$ be a set of* nanoκ *reaction rules.*

Given a presolution S , its encoding $[\![\mathsf{S}]\!]_\jmath$ is defined by $\jmath(\mathsf{S})$. The encoding of $\mathcal{R}$ is written $[\![\mathcal{R}]\!]_\jmath$ and is defined by $\{\rho_i : [\![\mathsf{L}_i]\!]_\jmath \overset{\lambda_i}{\rightarrow\!\!\!\!\triangleright} [\![\mathsf{R}_i]\!]_\jmath \mid i \in 1..n\}$.

Namely $[\![\mathcal{R}]\!]_\jmath$ and $[\![\mathsf{S}]\!]_\jmath$ are such that

- interfaces map sites to *gangs*, that is tuples of bonds of length n ;
- two distinct tuples do not contain the same name;
- tuples preserve the order of bonds in $\mathcal{R}$ and S.

The correctness of this part of the encoding is stated in Theorem 4.5.1.

4.3.2 From gangs to the nanoπ-calculus: agents as molecules.

The second step of our translation encodes given nanoκ systems with gangs of bonds into processes of nanoπ.

As discussed in section 4.1, we encode a species A by a parametric agent definition $\widehat{\mathtt{A}}(\widetilde{x}, \widetilde{y}, \widetilde{z}) = P$, whose parameters represent the possible values of fields and sites of the molecules of that species. The body P is a choice with a branch for every reaction involving the species A. A molecule $A[u](\sigma)$ is modeled by an invocation $\widehat{\mathtt{A}}(\{\!|u, \sigma|\!\})$.

We assume that the field and site names are totally ordered (using the lexicographic order for instance) so that we can index them with integers. Note that though the value of a field is an integer, it does not clash with the integer index, since the index only concerns the name of the field and not its value. We begin by defining $\{\!|u, \sigma|\!\}$. Let ε and $\neg\varepsilon$ be two distinguished channels. Then $\{\!|u, \sigma|\!\}$ is equal to the tuple $\{\!|u|\!\}_0, \{\!|\sigma|\!\}_1, \{\!|\sigma|\!\}_2$, where

- $\{\!|u|\!\}_0$ yields the tuple of the values of the fields in u;
- $\{\!|\sigma|\!\}_1$ yields the concatenation of the gangs in the range of σ;

- $\{\!|\sigma|\!\}_2$ yields a tuple of length equal to the size of $dom(\sigma)$, whose i-th element is ϵ if all the element of the tuple $\sigma(i)$ are ε and $\neg\epsilon$ if not.

The parameters in $\{\!|\sigma|\!\}_2$ permit us to check in the agent definition whether a site is free without using mismatch (see $[.]_\sigma$ in the 4-th item of the list below). Then we continue with a sequence of definitions, where n is the number of reactions in the nanoκ system considered:

- if $A[u](\sigma),B[v](\phi) \stackrel{\lambda}{\rightarrow} A[u'](\sigma'),\mathsf{S} \in \mathcal{R}$ then both $A[u](\sigma) \stackrel{\lambda}{\rightarrow} A[u'](\sigma') \in \mathcal{R}_L$ and $B[v](\phi) \stackrel{\lambda}{\rightarrow} \mathsf{S} \in \mathcal{R}_R$;
- If ρ is a creation, we define $\mathsf{CR}(\rho,\mathcal{R})$ to be the sequence $(x_1 : \lambda_1, \cdots, x_m : \lambda_m)$ where every subsequence $(x_{i\times n} : \lambda_{i\times n}, x_{i\times n+1} : \lambda_{i\times n+1}, \cdots, x_{i\times n+n-1} : \lambda_{i\times n+n-1})$ corresponds to the i-th bond created by ρ and $\lambda_{i\times n}, \cdots, \lambda_{i\times n+n-1}$ are the rates of the reactions in $\mathcal{R}$. Similarly if ρ is an exchange $\mathsf{EX}(\rho,\mathcal{R})$ is the set of the bonds exchanged by the reaction ρ.
- $[x_1, \cdots, x_m]_u$ is the sequence of matches $[x_i = u(i)]_{i\in\mathrm{dom}(u)}$ where u is a possibly partial evaluation;
- $[x_1, \cdots, x_m]_\sigma$ is the sequence of matches $(M_i)_{i\in\mathrm{dom}(\sigma)}$ where $M_i = [x_i = \epsilon]$ if all the element of the tuple $\sigma(i)$ are ε, and $M_i = [x_i = \neg\epsilon]$ if not, and where σ is a possibly partial interface;
- Given a tuple $\widetilde{x}$ of length m and a possibly partial valuation u whose domain has size m, $set_0(\widetilde{x}, u)$ is the tuple of length m whose i-th element is $u(i)$ whenever $i \in \mathrm{dom}(u)$, and the i-th element of $\widetilde{x}$, otherwise;
- Given two tuples $\widetilde{x}$ and $\widetilde{y}$ of size $m * n$ and $p * n$ respectively and two possibly partial interfaces σ and σ' that have the same domain of size m, $set_1(\widetilde{x}, \sigma, \sigma', \widetilde{y})$ is the tuple $(u_k)_{k\in[\![0..m*n-1]\!]}$ such that for all $i \in [\![0..m-1]\!]$ and $j \in [\![0..n-1]\!]$ the element u_{n*i+j} equals:
  - x_{n*i+j} if $i \notin \mathrm{dom}(\sigma)$ or $\sigma(i) = \sigma'(i)$ (i.e. i is neither mentioned nor affected by the interfaces);
  - ε if $\sigma(i) \in \mathrm{ran}(\sigma) \setminus \mathrm{ran}(\sigma')$ (i.e i is free in σ' but not in σ);
  - y_{n*k+j} if $\sigma'(i)$ is the k-th bond in $\mathrm{ran}(\sigma') \setminus \mathrm{ran}(\sigma)$ (i.e. it is the k-th bond created or exchanged);
- Given a tuple $\widetilde{x}$ of length m and a possibly partial interface σ whose domain has size m, $set_2(\widetilde{x}, \sigma)$ is the tuple of length m whose i-th element is x_i if $i \notin \mathrm{dom}(\sigma)$, otherwise it is ε if $\sigma(i) = \varepsilon^n$, and it is $\neg\varepsilon$ if $\sigma(i) \neq \varepsilon^n$;
- $proj(\widetilde{x}, a)$ is the tuple $(x_{n*a+i})_{i\in[\![0..n-1]\!]}$ and $proj(\widetilde{x}, a, i)$ is x_{n*a+i};

Intuitively the matches $[\widetilde{x}]_u$ and $[\widetilde{z}]_\sigma$, where $\widetilde{x}$ and $\widetilde{z}$ are expected to be some $\{\!|v|\!\}_0$ and $\{\!|\phi|\!\}_2$ respectively, check that $\widetilde{x}$ and $\widetilde{z}$ are compatible with requirements of u and σ respectively. The update function $set_0(\widetilde{x}, u)$, where

$\widetilde{x}$ is expected to be some $\{\!|v|\!\}_0$, updates $\widetilde{x}$ according to the valuation u. The update function $set_1(\widetilde{y}, \sigma, \sigma', \widetilde{w})$, where $\widetilde{y}$ and $\widetilde{w}$ are expected to be some $\{\!|\phi|\!\}_1$ and the names received from the synchronization respectively, updates $\widetilde{y}$ with the names of $\widetilde{w}$ or with ε according to the change from σ to σ'. The update function $set_2(\widetilde{z}, \sigma)$, where $\widetilde{z}$ is expected to be some $\{\!|\phi|\!\}_2$, updates $\widetilde{z}$ according to σ. The operation $proj(\widetilde{y}, a)$, where $\widetilde{y}$ is expected to be some $\{\!|\phi|\!\}_1$, extracts from $\widetilde{y}$ the gang corresponding to the site a, and the operation $proj(\widetilde{y}, a, i)$ extract from this gang the channel name corresponding to the reaction of index i.

Every preliminary notation is in place for the definition of the encoding from nanoκ with gangs to nanoπ.

Definition 4.3.2 *Given a* nanoκ *system, let $\mathcal{R}$ be its set of reactions and n be the size of $\mathcal{R}$. The* nanoπ *agent corresponding to the species A is:*

$$\begin{array}{rcl} \widehat{\mathtt{A}}(\widetilde{x}, \widetilde{y}, \widetilde{z}) & = & \sum_{\rho : A[u](\sigma) \stackrel{\lambda}{\to} A[u'](\sigma') \ \in \mathcal{R}_{\mathrm{L}}} [\widetilde{x}]_u \, [\widetilde{z}]_\sigma \, \alpha_{\rho,L} \, . \, P_{\rho,L} \\ & & + \sum_{\rho : A[u](\sigma) \stackrel{\lambda}{\to} \mathsf{S} \ \in \mathcal{R}_{\mathrm{R}}} [\widetilde{x}]_u \, [\widetilde{z}]_\sigma \, \alpha_{\rho,R} \, . \, P_{\rho,R} \end{array}$$

where the length of $\widetilde{x}$ is the number of fields of the species A, the length of $\widetilde{y}$ is n times the number of sites of A and the length $\widetilde{z}$ is the number of sites of A. In addition:

- *if ρ is a creation with an empty set of bonds in the left-hand side then $\alpha_{\rho,L} = \overline{\rho}(\widetilde{u : \lambda})$ and $\alpha_{\rho,R} = \rho\,(\widetilde{u})$ and $(\widetilde{u : \lambda}) = \mathsf{CR}(\rho, \mathcal{R})$;*
- *if ρ is a creation with a bond x in the left-hand side then $\alpha_{\rho,L} = \overline{proj(\widetilde{y}, a, i)}\,(\widetilde{u : \lambda})$ and $\alpha_{\rho,R} = proj(\widetilde{y}, a, i)\,(\widetilde{u})$, where a is the site of A bond by x, i is the index of ρ in $\mathcal{R}$ and $(\widetilde{u : \lambda}) = \mathsf{CR}(\rho, \mathcal{R})$;*
- *if ρ is a destruction with a bond x in the left-hand side then $\alpha_{\rho,L} = \overline{proj(\widetilde{y}, a, i)}\,(\ )$ and $\alpha_{\rho,R} = proj(\widetilde{y}, a, i)\,(\ )$, where a is the site of A bond by x and i is the index of ρ in $\mathcal{R}$;*
- *if ρ is an exchange with an empty set of bonds in the left-hand side or with a bond occurring once and in A then $\alpha_{\rho,L} = \overline{\rho}\,\widetilde{u}$ and $\alpha_{\rho,R} = \rho\,(\widetilde{u})$, where $\widetilde{u}$ is either empty, if there is no bond in the left-hand side, or $proj(\widetilde{y}, a)$ if the site with the bond is a;*
- *if ρ is an exchange with a bond x shared by the reactants then one defines $\alpha_{\rho,L} = \overline{proj(\widetilde{y}, a, i)}\,(\widetilde{u})$ and $\alpha_{\rho,R} = proj(\widetilde{y}, a, i)\,(\widetilde{v})$, where a is the site of A bond by x, i is the index of ρ in $\mathcal{R}$, $\widetilde{v}$ is a tuple of fresh names and $\widetilde{u}$ is either empty, if there is no bond in the left-hand side apart x, or $proj(\widetilde{y}, A, a')$ if A has a further bond on the site a'.*

As regards continuations:

- $P_{\rho,L} = \widehat{\mathtt{A}}(set_0(\widetilde{x}, u'), set_1(\widetilde{y}, \sigma, \sigma', \mathtt{bn}(\alpha_{\rho,L})), set_2(\widetilde{z}, \sigma'))$
- *if $\mathsf{S} = \_$ then $P_{\rho,R}$ is $\mathbf{0}$,*

- *if* $\mathsf{S} = A[u'](\sigma')\,,C_1[v_1](\phi_1)\,,\cdots,C_n[v_n](\phi_n)$ *then* $P_{\rho,R}$ *is* $\widehat{\mathtt{A}}(set_0(\widetilde{x},u'),$ $set_1(\widetilde{y},\sigma,\sigma',\mathtt{bn}(\alpha_{\rho,L})),set_2(\widetilde{z},\sigma'))\,,\widehat{\mathtt{C}_1}(\{\!|u_1|\!\}_1,\{\!|\phi_1|\!\}_2,\{\!|\phi_1|\!\}_3)\,,\cdots,$ $\widehat{\mathtt{C}_\mathtt{h}}(\{\!|u_h|\!\}_1,\{\!|\phi_h|\!\}_2,\{\!|\phi_h|\!\}_3)$.

Finally the encoding of a nanoκ solution with gangs is:

$$\{\!|A_1[u_1](\sigma_1)\,,\cdots,A_m[u_m](\sigma_m)|\!\} \triangleq (\delta_\mathsf{S})(\widehat{\mathtt{A}_1}\{\!|u_1,\sigma_1|\!\}\,,\cdots,\widehat{\mathtt{A}_\mathtt{m}}\{\!|u_m,\sigma_m|\!\})$$

where δ_S is the minimal set that contains

- $(\rho:\lambda)$, if ρ has no bond between its reactants and has rate λ,
- $(x_{(i,a),1}:\lambda_1,\ldots,x_{(i,a),n}:\lambda_n)$ if the site a of the molecule $A_i[u_i](\sigma_i)$ is bound, if $\widetilde{x_{(i,a)}}$ are the names of the corresponding gang, and where $(\lambda_1,\ldots,\lambda_n)$ are the rates of the reactions in $\mathcal{R}$.

Remarque 6 *Remarkably, our encoding retains some interesting compositional properties. First of all, consider two solutions* S *and* S' *and their encodings* $(\delta_\mathsf{S})P$ *and* $(\delta_{\mathsf{S}'})P'$*, respectively. Without loss of generality, let us suppose that* $\delta_\mathsf{S}\bigcap\delta_{\mathsf{S}'} = \mathtt{fn}(P)\bigcap\delta_{\mathsf{S}'} = \delta_\mathsf{S}\bigcap\mathtt{fn}(P') = \emptyset$*. This property can be guaranteed using* α*-conversion. Then the encoding of the parallel composition* $\mathsf{S}|\mathsf{S}'$ *is* $(\delta_\mathsf{S}\cup\delta_{\mathsf{S}'})(P|P')$ *where* $(\delta_\mathsf{S}\cup\delta_{\mathsf{S}'})$ *is the union of the name declarations* δ_S *and* $\delta_{\mathsf{S}'}$*. More interestingly, if we add a new reaction rule to a system that we have already encoded, there are very few changes to be made. First, we need to expand the length of the gangs by one. Then, we have to add one line to the definitions corresponding to the two species occurring in the left hand side of the new reaction, in order to describe this additional behavior for the molecules belonging to these two species.*

4.4 The encoding at work.

To illustrate our encoding we chose a toy-modeling of the transcription of a gene. There are three species:

- *Gn* models a gene. It has one field *tr* and two sites *pr* and *rnap*; *tr* is 1 when the gene is being transcribed by the RNA polymerase and 0 if not; *pr* and *rnap* are used to link to *Pr* and *RNAp*, respectively;
- *Pr* models the various promoter-sequences of the gene. It has one field *act* and two sites *rnap* and *gn*. The activation of the promoters by the transcription-factors is represented by switching *act* from 0 to 1. The sites *rnap* and *gn* are used to link *RNAp* and *Gn*, respectively;
- *RNAp* models the RNA polymerase. It has one field *act* that is set to 1 when the molecule is activated by the complexations with the promoters and the transcription factors, it is set to 0 otherwise. It has a site *link* that may be bound either to *Pr* or to *Gn*, according to the stage of the transcription;

There are three reactions. The creation ρ_1 models the binding of the RNA polymerase to the promoters (between sites *rnap* of *Pr* and *link* of *RNAp*) and their activation (update of the fields *act* of *Pr* and *RNAp*). The exchange ρ_2 models the motion of the RNA-polymerase to the gene and the beginning of the transcription itself (update of the field *tr* of *Gn* to 1). The destruction ρ_3 models the termination of the transcription. For the sake of the simplicity we chose not to model the release of a messenger RNA.

$$\rho_1 :\ Pr[act^0](rnap), RNAp[act^0](link) \overset{\lambda_1}{\to} Pr[act^1](rnap^x), RNAp[act^1](link^x)$$

$$\rho_2 :\ Pr[act^1](rnap^y + gn^x)\,,\ Gn[tr^0](pr^x + rnap) \overset{\lambda_2}{\to} Pr[act^0](rnap + gn^x)\,,\ Gn[tr^1](pr^x + rnap^y)$$

$$\rho_3 :\ RNAp[act^1](link^x)\,,\ Gn[tr^1](rnap^x) \overset{\lambda_3}{\to} RNAp[act^0](link)\,,\ Gn[tr^0](rnap)$$

The encoding of this `nano`κ system yields the following three recursive definitions in `nano`π. We notice that, in the encoding of *Gn*, the parameters $p\rho_1, p\rho_2, p\rho_3, ?p$ correspond to the gang of the site *pr* (the parameters $p\rho_1, p\rho_2, p\rho_3$ are yielded by $\{\![.]\!\}_1$ and the parameter $?p$ by $\{\![.]\!\}_2$), similarly the parameters $r\rho_1, r\rho_2, r\rho_3, ?r$ correspond to the gang of the site *rnap*.

$$\begin{aligned}&\widehat{\texttt{Gn}}(tr, p\rho_1, p\rho_2, p\rho_3, r\rho_1, r\rho_2, r\rho_3, ?p, ?r) \triangleq\\ &\quad [tr = 0, ?p = \neg\epsilon, ?r = \epsilon]\, p\rho_2\,(r_1, r_2, r_3)\,.\,\widehat{\texttt{Gn}}(1, p\rho_1, p\rho_2, p\rho_3, r_1, r_2, r_3, ?p, \neg\epsilon)\\ &\quad + [tr = 1, ?r = \neg\epsilon]\, r\rho_3\,()\,.\,\widehat{\texttt{Gn}}(1, p\rho_1, p\rho_2, p\rho_3, \epsilon, \epsilon, \epsilon, ?p, \epsilon)\end{aligned}$$

In the encoding of *Pr*, the parameters $r\rho_1, r\rho_2, r\rho_3, ?r$ correspond to the gang of the site *rnap* and the parameters $g\rho_1, g\rho_2, g\rho_3, ?g$ correspond to the gang of the site *gn*.

$$\begin{aligned}&\widehat{\texttt{Pr}}(act, r\rho_1, r\rho_2, r\rho_3, g\rho_1, g\rho_2, g\rho_3, ?r, ?g) \triangleq\\ &\quad [act = 0, ?r = \epsilon]\, \overline{\rho_1}\,(r_1 : \lambda_1, r_2 : \lambda_2, r_3 : \lambda_3)\,.\,\widehat{\texttt{Pr}}(1, r_1, r_2, r_3, g\rho_1, g\rho_2, g\rho_3, \neg\epsilon, ?g)\\ &\quad + [act = 1, ?r = \neg\epsilon, ?g = \neg\epsilon]\, \overline{g\rho_2}\,(r\rho_1, r\rho_2, r\rho_3)\,.\,\widehat{\texttt{Pr}}(0, \epsilon, \epsilon, \epsilon, g\rho_1, g\rho_2, g\rho_3, \epsilon, ?g)\end{aligned}$$

In the encoding of the species *RNAp*, the parameters $l\rho_1, l\rho_2, l\rho_3, ?l$ correspond to the gang of the site *link*.

$$\begin{aligned}&\widehat{\texttt{RNAp}}(act, l\rho_1, l\rho_2, l\rho_3, ?l) \triangleq\\ &\qquad [act = 0, ?l = \epsilon]\, \rho_1\,(r_1, r_2, r_3))\,.\,\widehat{\texttt{RNAp}}(1, r_1, r_2, r_3, \neg\epsilon)\\ &\qquad + [act = 1, ?l = \neg\epsilon]\, \overline{l\rho_3}\ .\widehat{\texttt{RNAp}}(0, \epsilon, \epsilon, \epsilon, \epsilon)\end{aligned}$$

Finally, the encoding of the solution $Pr[act^0](rnap + gn^x)$, $Gn[tr^0](prev^x + rnap)$, $RNAp[act](link)$ is the term

$$\begin{aligned}(\rho_1 : \lambda_1, pr_1 : \lambda_1, pr_2 : \lambda_2, pr_3 : \lambda_3)(\quad &\widehat{\texttt{Pr}}(0, \epsilon, \epsilon, \epsilon, pr_1, pr_2, pr_3, \epsilon, \neg\epsilon)\\ &\mid \widehat{\texttt{Gn}}(0, pr_1, pr_2, pr_3, \epsilon, \epsilon, \epsilon, \neg\epsilon, \epsilon)\\ &\mid \widehat{\texttt{RNAp}}(act, \epsilon, \epsilon, \epsilon, \epsilon)\\ )\end{aligned}$$

4.5 Correctness of the encoding.

In this section we state and prove the correctness of our encoding. We first prove in Theorem 4.5.1 that the first step of the encoding provides a one-to-one correspondence between the basic transitions of the terms of the nanoκ calculus and the basic transitions of the nanoκ calculus with gangs. Similarly we prove in Theorem 4.5.2 that the second step of the encoding provides a one-to-one correspondence between the basic transitions of the terms of the nanoκ calculus with gangs and the basic transitions of the nanoπ-calculus. Then in Theorem 4.5.3 we prove how to lift the correctness from the basic transition relation to the collective transition relation. And finally in Theorem 4.5.4 we state the correctness of the full encoding with respect to the collective transition relation.

4.5.1 Correctness of the encoding from nanoκ to nanoκ with gangs with respect to the basic transition relation.

Before stating the correctness of the introduction of the gangs we need a few definitions. Given a renaming $\imath$, we define $[\![\imath]\!]_\jmath$ to be the renaming which coincides with $\jmath \circ \imath \circ \jmath^{-1}$ pointwise (recall that $\jmath$ is the bijective function associating gangs to names in the definition 4.3.1). We have that if $\imath$ is injective and order-preserving then $[\![\imath]\!]_\jmath$ is also injective and order-preserving because $\jmath$ is bijective and such that $x \leq y$ if and only if $\jmath(x) \leq \jmath(y)$. Then for $\alpha \in \{L, R\}$ we define $[\![\rho_\alpha, \imath]\!]_\jmath \triangleq ([\![\rho]\!]_\jmath)_\alpha, [\![\imath]\!]_\jmath$.

For the sake of the readability, from now on we omit the subscript $\jmath$ in the encoding and write only $[\![.]\!]$.

We can now state the correctness of the introduction of gangs with respect to the basic transition relation:

Theorem 4.5.1 *For all solutions* S *and* T *of the* nanoκ *calculus:*

1. *if* $\mathsf{S} \xrightarrow{\mu}_l \mathsf{T}$ *then* $[\![\mathsf{S}]\!] \xrightarrow{[\![\mu]\!]}_l [\![\mathsf{T}]\!]$.
2. *if* $\mathsf{S} \xrightarrow{\rho}_{l,l'} \mathsf{T}$ *then* $[\![\mathsf{S}]\!] \xrightarrow{[\![\rho]\!]}_{l,l'} [\![\mathsf{T}]\!]$.
3. *if* $[\![\mathsf{S}]\!] \xrightarrow{\mu}_l \mathsf{T}$ *then there exists* S' *and* μ' *such that:* $[\![\mathsf{S}']\!] = \mathsf{T}$, $\mathsf{S} \xrightarrow{\mu'}_l \mathsf{S}'$, *and* $\mu = [\![\mu']\!]$.
4. *if* $[\![\mathsf{S}]\!] \xrightarrow{\rho}_{l,l'} \mathsf{T}$ *then there exists* S' *and* ρ' *such that:* $[\![\mathsf{S}']\!] = \mathsf{T}$, $\mathsf{S} \xrightarrow{\rho'}_{l,l'} \mathsf{S}'$, *and* $\rho = [\![\rho']\!]$.

Proof

1. We prove the result by induction on the derivation tree of $\mathsf{S} \xrightarrow{\mu}_1 \mathsf{T}$.

- If the transition $\mathsf{S} \xrightarrow{\mu}_l \mathsf{T}$ has been obtained by the (init) rule, there is a rule $\rho : A[u](\sigma), B[v](\phi) \overset{\lambda}{\nrightarrow} A[u'](\sigma'), \mathsf{R}$ such that the transition is written:

  (α) either $A[u+w](\imath \circ \sigma + \nu) \xrightarrow{\rho_L, \imath}_1 A[u'+w](\imath \circ \sigma' + \nu)$

  (β) or $B[v+w](\imath \circ \phi + \nu) \xrightarrow{\rho_R, \imath}_1 \imath(\mathsf{R})$

  with $\mathrm{ran}(\imath) \bigcap \mathrm{ran}(\nu) = \emptyset$. The reaction ρ is translated into $[\![\rho]\!] : A[u](\jmath \circ \sigma), B[v](\jmath \circ \phi) \overset{\lambda}{\nrightarrow} A[u'](\jmath \circ \sigma'), \jmath(\mathsf{S})$. So using the (init) rule one can derive respectively:

  (α) either $A[u+w](\imath' \circ \jmath \circ \sigma + \nu') \xrightarrow{[\![\rho]\!]_L, \imath'}_1 A[u'+w](\imath' \circ \jmath \circ \sigma' + \nu')$

  (β) or $B[v+w](\imath' \circ \jmath \circ \phi + \nu') \xrightarrow{[\![\rho]\!]_R, \imath'}_1 (\imath' \circ \jmath)(\mathsf{R})$

  assuming that $\mathrm{ran}(\imath') \bigcap \nu' = \emptyset$. We can choose $\imath' = [\![\imath]\!]$ and $\nu' = \jmath \circ \nu$ because $\mathrm{ran}([\![\imath]\!]) \bigcap \jmath \circ \nu = \emptyset$. Indeed $\mathrm{ran}([\![\imath]\!]) \bigcap \jmath \circ \nu' = \emptyset$ if and only if $\mathrm{ran}(\imath \circ \jmath^{-1}) \bigcap \nu = \emptyset$ since $\jmath$ is bijective, and $\mathrm{ran}(\imath \circ \jmath^{-1}) \bigcap \nu = \emptyset$ is implied by $\mathrm{ran}(\imath) \bigcap \nu = \emptyset$. Now since $[\![\imath]\!] \circ \jmath = \jmath \circ \imath$ we deduce respectively:

  (α) either $[\![A[u+w](\imath \circ \sigma + \nu)]\!] \xrightarrow{[\![\rho_L, \imath]\!]}_1 [\![A[u'+w](\imath \circ \sigma' + \nu)]\!]$

  (β) or $[\![B[v+w](\imath \circ \phi + \nu)]\!] \xrightarrow{[\![\rho_R, \imath]\!]}_1 [\![\imath(\mathsf{R})]\!]$

  which concludes the proof of this case.

- If the transition $S \xrightarrow{\mu}_l T$ has been obtained by the (lift) rule, there are S_1, S_2 and S' such that $bn(\mu) \bigcap \mathrm{name}(S') = \emptyset$ and either $S_1 \xrightarrow{\mu}_l S_2$ and $S = S_1, S'$ and $T = S_2, S'$ or $S_1 \xrightarrow{\mu}_{l-\#(\mathsf{S}')} S_2$ and $S = S', S_1$ and $T = S', S_2$, where $\#(\mathsf{S}')$ is the length of S'.

  If we are in the former case, then by induction hypothesis we have that $[\![S_1]\!] \xrightarrow{[\![\mu]\!]}_l [\![S_2]\!]$. Moreover since $\jmath$ is bijective and $(\mathrm{name}(\mathsf{S}_2) \setminus \mathrm{name}(\mathsf{S}_1)) \bigcap n(S') = \emptyset$ we have that $(\mathrm{name}([\![\mathsf{S}_2]\!]) \setminus \mathrm{name}([\![\mathsf{S}_1]\!])) \bigcap \mathrm{name}([\![S']\!]) = \emptyset$. So by the (lift) rule we obtain $[\![S]\!] = [\![S_1]\!], [\![S']\!] \xrightarrow{[\![\mu]\!]}_l [\![S_2]\!], [\![S']\!] = [\![T]\!]$.

  In the other case, the proof is achieved with the same arguments.

2. Suppose that $\mathsf{S} \xrightarrow{\rho}_{l,l'} \mathsf{T}$. This transition has been obtained with the (communication) rule, so there are $\mathsf{S_1}$, $\mathsf{S_2}$, $\mathsf{T_1}$, $\mathsf{T_2}$, μ and $\imath$ such that $S = S_1, S_2$ and $T = \imath(T_1, T_2)$ and $\mathsf{S_1} \xrightarrow{\mu}_l \mathsf{T_1}$ and $\mathsf{S_2} \xrightarrow{\overline{\mu}}_{l'-\#(\mathsf{S_1})} \mathsf{T_2}$ where $\#(\mathsf{S_1})$ is the size of $\mathsf{S_1}$ and where $\imath$ maps $\mathrm{name}(\mathsf{T_1}, \mathsf{T_2}) \setminus \mathrm{name}(\mathsf{S})$ to the least names not occurring in S. By the first item of the theorem we obtain that $[\![\mathsf{S_1}]\!] \xrightarrow{[\![\mu]\!]}_l [\![\mathsf{T_1}]\!]$ and $[\![\mathsf{S_2}]\!] \xrightarrow{[\![\overline{\mu}]\!]}_{l'-\#(\mathsf{S_1})} [\![\mathsf{T_2}]\!]$.

  Since $[\![\overline{\mu}]\!] = \overline{[\![\mu]\!]}$ and $\#(S_1) = \#([\![S_1]\!])$, by the (communication) rule we obtain that: $[\![\mathsf{S_1}]\!], [\![\mathsf{S_2}]\!] \xrightarrow{[\![\rho]\!]}_{l,l'} \imath'([\![\mathsf{T_1}]\!], [\![\mathsf{T_2}]\!])$ for some $\imath'$ mapping $\mathrm{name}([\![\mathsf{T_1}]\!], [\![\mathsf{T_2}]\!]) \setminus \mathrm{name}([\![\mathsf{S_1}]\!], [\![\mathsf{S_2}]\!])$ to the least names not occurring in $[\![\mathsf{S_1}]\!], [\![\mathsf{S_2}]\!]$, that is in $[\![\mathsf{S}]\!]$.

Because $\jmath$ is injective and such that $x \leq y \Leftrightarrow \jmath(x) \leq \jmath(y)$ we have that $[\![\imath]\!]$ maps $\text{name}([\![\mathsf{T}_1]\!], [\![\mathsf{T}_2]\!]) \setminus \text{name}([\![\mathsf{S}_1]\!], [\![\mathsf{S}_2]\!])$ to the least names not occurring in $[\![\mathsf{S}_1]\!], [\![\mathsf{S}_2]\!]$, that is in $[\![\mathsf{S}]\!]$. So we can choose $\imath' = [\![\imath]\!]$. Since $[\![\imath]\!] \circ \jmath = \jmath \circ \imath$ we have that $[\![\imath]\!]([\![\mathsf{T}_1]\!], [\![\mathsf{T}_2]\!]) = ([\![\imath]\!] \circ \jmath)(\mathsf{T}_1, \mathsf{T}_2) = (\jmath \circ \imath)(\mathsf{T}_1, \mathsf{T}_2) = [\![T]\!]$. And thus: $[\![\mathsf{S}]\!] \xrightarrow{[\![\rho]\!]}_{l,l'} [\![\mathsf{T}]\!]$.

3. We prove the result by induction on the derivation tree of $[\![\mathsf{S}]\!] \xrightarrow{\mu}_l T$.

   - If the transition $[\![S]\!] \xrightarrow{\mu}_l T$ has been obtained by the (init) rule, there is a rule $[\![\rho]\!] : A[u](\jmath \circ \sigma), B[v](\jmath \circ \phi) \xrightarrow{\lambda} A[u'](\jmath \circ \sigma'), \jmath(\mathsf{R})$ such that the transition is written:

     (α) either $A[u + w](\imath \circ \jmath \circ \sigma + \nu) \xrightarrow{[\![\rho]\!]_L, \imath}_1 A[u' + w](\imath \circ \jmath \circ \sigma' + \nu)$

     (β) or $B[v + w](\imath \circ \jmath \circ \phi + \nu) \xrightarrow{[\![\rho]\!]_R, \imath}_1 (\imath \circ \jmath)(\mathsf{R})$

     with $\text{ran}(\imath) \bigcap \text{ran}(\nu) = \emptyset$. So the reaction ρ is $A[u](\sigma), B[v](\phi) \xrightarrow{\lambda} A[u'](\sigma'), \mathsf{R}$. So using the (init) rule we can derive respectively:

     (α) either $A[u + w](\imath' \circ \sigma + \nu') \xrightarrow{\rho_L, \imath'}_1 A[u' + w](\imath' \circ \sigma' + \nu')$

     (β) or $B[v + w](\imath' \circ \phi + \nu') \xrightarrow{\rho_R, \imath'}_1 \imath(\mathsf{R})$

     with $\text{ran}(\imath') \bigcap \text{ran}(\nu') = \emptyset$. We can choose $\imath' = \jmath^{-1} \circ \imath \circ \jmath$ and $\nu' = \jmath^{-1} \circ \nu$ because $\text{ran}(\jmath^{-1} \circ \imath \circ \jmath) \bigcap \jmath^{-1} \circ \nu = \emptyset$. Indeed $\text{ran}(\jmath^{-1} \circ \imath \circ \jmath) \bigcap \jmath^{-1} \circ \nu = \emptyset$ if and only if $\text{ran}(\imath \circ \jmath) \bigcap \nu = \emptyset$ since $\jmath$ is bijective, and $\text{ran}(\imath \circ \jmath) \bigcap \nu = \emptyset$ is implied by $\text{ran}(\imath) \bigcap \nu = \emptyset$. Now it suffices to define respectively:

     (α) either $S' \triangleq A[u' + w](\jmath^{-1} \circ \imath \circ \jmath \circ \sigma' + \jmath^{-1} \circ \nu)$ and $\mu' \triangleq \rho_L, \jmath^{-1} \circ \imath \circ \jmath$

     (β) or $S' \triangleq (\jmath^{-1} \circ \imath)(\mathsf{R})$ and $\mu' \triangleq \rho_R, \jmath^{-1} \circ \imath \circ \jmath$

     to conclude the proof of this case.

   - If the transition $[\![\mathsf{S}]\!] \xrightarrow{\mu}_l \mathsf{T}$ has been obtained by the (lift) rule, there are S_1, T' and S_2 such that $(\text{name}([\![\mathsf{S}_1]\!]) \setminus \text{name}(T)) \bigcap \text{name}([\![\mathsf{S}_2]\!]) = \emptyset$ and either $[\![\mathsf{S}]\!] = [\![\mathsf{S}_1]\!], [\![\mathsf{S}_2]\!]$ and $\mathsf{T} = \mathsf{T}', [\![\mathsf{S}_2]\!]$ and $[\![\mathsf{S}_1]\!] \xrightarrow{\mu}_l \mathsf{T}'$ or $\mathsf{S} = [\![\mathsf{S}_2]\!], [\![\mathsf{S}_1]\!]$ and $T = [\![\mathsf{S}_2]\!], \mathsf{T}'$ and $[\![\mathsf{S}_1]\!] \xrightarrow{\mu}_{l - \#([\![\mathsf{S}_2]\!])} \mathsf{T}'$, where $\#([\![\mathsf{S}_2]\!])$ is the length of $[\![\mathsf{S}_2]\!]$.

     If we are in the former case, then by induction hypothesis we have that there exists S'' and μ' such that $[\![S'']\!] = T'$ and $S_1 \xrightarrow{\mu'}_l S''$ and $\mu = [\![\mu']\!]$. Moreover since $\jmath$ is bijective and $(\text{name}([\![\mathsf{S}_1]\!]) \setminus \text{name}(T)) \bigcap \text{name}([\![S_2]\!]) = \emptyset$ we have that $(\text{name}(\mathsf{S}_1) \setminus \text{name}(\mathsf{S}'')) \bigcap n(S_2) = \emptyset$. So by the (lift) rule we obtain $[\![S]\!] = [\![S_1]\!], [\![S_2]\!] \xrightarrow{[\![\mu']\!]}_l [\![S'']\!], [\![S_2]\!] = T$.

     In the other case, the proof is achieved with the same arguments.

4. Suppose that $[\![\mathsf{S}]\!] \xrightarrow{\rho}_{l,l'} \mathsf{T}$. This transition has been obtained with the (communication) rule, so there are $[\![\mathsf{S}_1]\!]$, $[\![\mathsf{S}_2]\!]$, T_1, T_2, μ and $\imath$ such that $S = S_1, S_2$ and $T = \imath(T_1, T_2)$ and $[\![\mathsf{S}_1]\!] \xrightarrow{\mu}_l \mathsf{T}_1$ and $[\![\mathsf{S}_2]\!] \xrightarrow{\overline{\mu}}_{l' - \#([\![\mathsf{S}_1]\!])} \mathsf{T}_2$

where $\#(\llbracket \mathsf{S}_1 \rrbracket)$ is the size of $\llbracket \mathsf{S}_1 \rrbracket$, where ρ is the rule of μ and where $\imath$ maps $\mathrm{name}(\mathsf{S}_1, \mathsf{S}_2) \setminus \mathrm{name}(\mathsf{T}_1, \mathsf{T}_2)$ to the least names not occurring in $\llbracket \mathsf{S} \rrbracket$. By the third item of the theorem we obtain that there exists S'_1, S'_2 and μ such that $\llbracket \mathsf{S}'_1 \rrbracket = \mathsf{T}_1$ and $\llbracket \mathsf{S}'_2 \rrbracket = \mathsf{T}_2$ and $\mathsf{S}_1 \xrightarrow{\mu'} \mathsf{S}'_1$ and $\mathsf{S}_2 \xrightarrow{\overline{\mu'}} \mathsf{S}'_2$ and $\llbracket \mu' \rrbracket = \mu$.

Since $\#(\mathsf{S}_1) = \#(\llbracket \mathsf{S}_1 \rrbracket)$, by the (communication) rule we obtain that $\mathsf{S}_1, \mathsf{S}_2 \xrightarrow{\rho'}_{l,l'} \imath'(\mathsf{S}'_1, \mathsf{S}'_2)$ where $\llbracket \rho' \rrbracket = \rho$ and for some $\imath'$ mapping $\mathrm{name}(\mathsf{S}'_1, \mathsf{S}'_2)$ $s \setminus \mathrm{name}(\mathsf{S}_1, \mathsf{S}_2)$ to the least names not occurring in $\mathsf{S}_1, \mathsf{S}_2$, that is in S.

Because $\jmath$ is injective and such that $x \leq y \Leftrightarrow \jmath(x) \leq \jmath(y)$ we have that $\jmath^{-1} \circ \imath \circ \jmath$ maps $\mathrm{name}(\mathsf{S}'_1, \mathsf{S}'_2) \setminus \mathrm{name}(\mathsf{S}_1, \mathsf{S}_2)$to the least names not occurring in $\mathsf{S}_1, \mathsf{S}_2$, that is in S. So we can choose $\imath' = \jmath^{-1} \circ \imath \circ \jmath$. Now it suffices to define $S' = (\jmath^{-1} \circ \imath \circ \jmath)(\mathsf{S}'_1, \mathsf{S}'_2)$: we do have that $\llbracket \mathsf{S}' \rrbracket = \jmath((\jmath^{-1} \circ \imath \circ \jmath)(\mathsf{S}'_1, \mathsf{S}'_2)) = (\imath \circ \jmath)(\mathsf{S}'_1, \mathsf{S}'_2) = \imath(\mathsf{T}_1, \mathsf{T}_2) = \mathsf{T}$ and $\mathsf{S} \xrightarrow{\rho'}_{l,l'} \mathsf{S}'$ and $\llbracket \rho' \rrbracket = \rho$.

□

4.5.2 Correctness of the encoding from nanoκ with gangs to nanoπ with respect to the basic transition relation.

Before stating the correctness of the encoding in the nanoπ-calculus we need a few definitions. If A is the l-th molecule in S and if the rule of μ corresponds to the i-th branch of the choice in $\widehat{\mathtt{A}}$, we define $\{\!|l, S, \mu|\!\}$ to be the pair (l, i). We also let $\{\!|\rho_L, \imath|\!\}$ and $\{\!|\rho_R, \imath|\!\}$ to be respectively $\alpha_{\rho,L}$ and $\alpha_{\rho,R}$ as defined in Definition 4.3.2. We also define $\langle A_1[u_1](\sigma_1), \ldots, A_m[u_m](\sigma_m)\rangle \triangleq \widehat{A_1}\{\!|u_1, \sigma_1|\!\}, \ldots, \widehat{A_m}\{\!|u_m, \sigma_m|\!\}$ that is the encoding deprived of the name declarations.

Finally we define $\overset{\circ}{\equiv}_\alpha$ to be the α-renaming where names are only permuted. That is given two processes $S \overset{\circ}{\equiv}_\alpha T$ if and only if S is an α-renaming of T as defined in the structural congruence (see definition 4.2.2) and $\mathtt{bn}(S) = \mathtt{bn}(T)$.

Theorem 4.5.2 *1. If* $\mathsf{S} \xrightarrow{\mu}_l \mathsf{T}$ *then:*

- *if* $\mu = \rho_R, \imath$ *and* ρ *is an exchange then:*

$$\langle \mathsf{S} \rangle \xrightarrow{\{\!|\mu|\!\}}_{\{\!|l,S,\mu|\!\}} \langle \mathsf{T}\{\mathtt{bn}(\{\!|\mu|\!\})/\imath(\mathtt{EX}(\rho))\}\rangle$$

- *otherwise* $\langle \mathsf{S} \rangle \xrightarrow{\{\!|\mu|\!\}}_{\{\!|l,S,\mu|\!\}} \langle \mathsf{T} \rangle$.

2. If $\mathsf{S} \xrightarrow{\rho}_{l,l'} \mathsf{T}$ *then* $\{\!|\mathsf{S}|\!\} \xrightarrow{\tau_{rate(\rho)}}_{\{\!|l,S,\rho|\!\},\{\!|l',S,\rho|\!\}} \overset{\circ}{\equiv}_\alpha \{\!|\mathsf{T}|\!\}$.

3. If $\langle \mathsf{S} \rangle \xrightarrow{\mu}_{l.i} \mathsf{T}$ *then there exist* S' *and* μ' *such that* $\mathsf{S} \xrightarrow{\mu'}_l \mathsf{S}'$, $\mu = \{\!|\mu'|\!\}$ *and:*

- *if* $\mu' = \rho_R, \imath$ *and* ρ *is an exchange then* $\langle \mathsf{S}'\{\mathtt{bn}(\mu)/\imath(\mathtt{EX}(\rho))\}\rangle = \mathsf{T}$,
- *otherwise* $\langle \mathsf{S}' \rangle = \mathsf{T}$.

4. *If* $\{\![\mathsf{S}]\!\} \xrightarrow{\tau_\lambda}_{l.i,l'.i'} \mathsf{T}$ *then there exist* S' *and* ρ *such that:* $\{\![\mathsf{S}']\!\} \overset{\circ}{\equiv}_\alpha \mathsf{T}$, $\mathsf{S} \xrightarrow{\rho'}_{l,l'} \mathsf{S}'$, *and* $rate(\rho) = \lambda$.

Before proving Theorem 4.5.2, we gather some facts about our encoding in the following lemmas.

Lemma 4.5.1
- $[\{\![u]\!\}_0]_v$ *is true if and only if* $\exists v'.u = v + v'$.
- $[\{\![\sigma]\!\}_2]_\phi$ *is true if and only if* $\exists \imath, \nu.\sigma = \imath \circ \phi + \nu$.

Proof By definition of $[.]_v$, $[.]_\phi$ and $\{\![.]\!\}$. □

Lemma 4.5.2 *For any evaluation* u *and any interface* σ *we have that:*

- $set_0(\{\![u + w]\!\}_0, u') = \{\![u' + w]\!\}_0$ *if* $\mathrm{dom}(u) = \mathrm{dom}(u')$,
- *and* $set_2(\{\![\imath \circ \sigma + \nu]\!\}_2, \sigma') = \{\![\imath \circ \sigma' + \nu]\!\}_2$ *if* $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$.

Given a transition $\rho : A[u](\imath \circ \sigma + \nu) \xrightarrow{\mu} A[u'](\imath \circ \sigma' + \nu), \mathsf{S}$, *it holds that:*

- $set_1(\{\![\imath \circ \sigma + \nu]\!\}_1, \sigma, \sigma', \imath(\mathsf{CR}(\rho, \mathcal{R}))) = \{\![\imath \circ \sigma' + \nu]\!\}_1$ *if* ρ *is a creation,*
- $set_1(\{\![\imath \circ \sigma + \nu]\!\}_1, \sigma, \sigma', \emptyset) = \{\![\imath \circ \sigma' + \nu]\!\}_1$ *if* ρ *is a destruction,*
- *and* $set_1(\{\![\imath \circ \sigma + \nu]\!\}_1, \sigma, \sigma', \imath(\mathsf{EX}(\rho, \mathcal{R}))) = \{\![\imath \circ \sigma' + \nu]\!\}_1$ *if* ρ *is an exchange.*

And:

- $\langle A[u' + w](\imath \circ \sigma' + \nu), \mathsf{S}\rangle = \widehat{A}(set_0(\{\![u + w]\!\}_0, u'), set_1(\{\![\imath \circ \sigma + \nu]\!\}_1, \sigma, \sigma', \mathcal{B}), set_2(\{\![\imath \circ \sigma + \nu]\!\}_2, \sigma')), \langle \mathsf{S}\rangle$ *where* $\mathcal{B}$ *is either* $\imath(\mathsf{CR}(\rho, \mathcal{R}))$, $\emptyset$ *or* $\imath(\mathsf{EX}(\rho, \mathcal{R}))$ *if* ρ *is a creation, a destruction or an exchange respectively.*

Proof By definition of the *set* functions and $\{\![.]\!\}$. □

We are now ready to prove Theorem 4.5.2.

Proof (of Theorem 4.5.2)

1. We prove the result by induction on the derivation tree of $\mathsf{S} \xrightarrow{\mu}_l \mathsf{T}$.
   - If transition $\mathsf{S} \xrightarrow{\mu}_l \mathsf{T}$ has been obtained by the (init) rule, there is a rule $\rho : A[u](\sigma), B[v](\phi) \xrightarrow{\lambda} A[u'](\sigma'), \mathsf{R}$ such that the transition is written:
     - either $A[u + w](\imath \circ \sigma + \nu) \xrightarrow{\rho_L, \imath}_1 A[u' + w](\imath \circ \sigma' + \nu)$
     - or $B[v + w](\imath \circ \phi + \nu) \xrightarrow{\rho_R, \imath}_1 \imath(\mathsf{R})$

     with $\mathrm{ran}(\imath) \bigcap \mathrm{ran}(\nu) = \emptyset$.

     Let us suppose that we are in the second case and that R is not the empty solution. Otherwise the proof is achieved similarly.

In $\widehat{B}(\widetilde{x},\widetilde{y},\widetilde{z})$ the branch corresponding to ρ (let us call j its index) is written $[\widetilde{x}]_v[\widetilde{z}]_\phi\alpha_{\rho,R}.(\widehat{B}(set_0(\widetilde{x},v'),set_1(\widetilde{y},\phi,\phi',\mathtt{bn}(\alpha_{\rho,R})),set_2(\widetilde{z},\phi)),\langle\mathsf{R}'\rangle)$ where $\alpha_{\rho,R}$ is defined by the definition 4.3.2.

By Lemma 4.5.1 we have that the matches $[\{\!|v+w|\!\}]_v$ and $[\{\!|\imath\circ\phi+\nu|\!\}]_\phi$ are true. So by the (init) rule we obtain that $\widehat{B}(\{\!|v+w|\!\}_0,\{\!|\imath\circ\phi+\nu|\!\}_1,\{\!|\imath\circ\phi+\nu|\!\}_2) \xrightarrow{\alpha_{\rho,R}}_{1.j} \widehat{B}(set_0(\{\!|v+w|\!\}_0,v'),set_1(\{\!|\imath\circ\phi+\nu|\!\}_1,\sigma,\sigma',\mathtt{bn}(\alpha_{\rho,R})),set_2(\{\!|\imath\circ\phi+\nu|\!\}_2,\phi')),\langle\mathsf{R}'\rangle$.

The rest of the proof depends on μ:

- if $\mu=\rho_R,\imath$ and ρ is an exchange then by Lemma 4.5.2 we have that $\widehat{B}(set_0(\{\!|v+w|\!\}_0,v'),set_1(\{\!|\imath\circ\phi+\nu|\!\}_1,\phi,\phi',\mathtt{bn}(\alpha_{\rho,R})),set_2(\{\!|\imath\circ\phi+\nu|\!\}_2,\phi'),\langle\mathsf{R}'\rangle)=\langle B[v'+w](\imath\circ\phi'+\nu)\{\mathtt{bn}(\alpha_{\rho,R})/\imath(\mathtt{EX}(\rho))\},\mathsf{R}'\rangle$, since in $\mathtt{nano}\kappa$ the exchanged names are already passed to the products of the reaction at the step of the (init) rule, while in $\mathtt{nano}\pi$ it is only achieved by the communication during the step of the (communication) rule.
- otherwise by Lemma 4.5.2 we have that $\widehat{B}(set_0(\{\!|v+w|\!\}_0,v'),set_1(\{\!|\imath\circ\phi+\nu|\!\}_1,\phi,\phi',\mathtt{bn}(\alpha_{\rho,R})),set_2(\{\!|\imath\circ\phi+\nu|\!\}_2,\phi'),\langle\mathsf{R}'\rangle)=\langle B[v'+w](\imath\circ\phi'+\nu),\mathsf{R}'\rangle$.

It suffices now to notice that $\{\!|\rho_R,\imath|\!\}=\alpha_{\rho,R}$ and $1.j=\{\!|l,\mathsf{S},\mu|\!\}$.

- If the transition $S\xrightarrow{\mu}_l T$ has been obtained by the (lift) rule, there are S_1, S_2 and S' such that either $S_1\xrightarrow{\mu}_l S_2$ and $S=S_1,S'$ and $T=S_2,S'$ or $S_1\xrightarrow{\mu}_{l-\#(\mathsf{S}')}S_2$ and $S=S',S_1$ and $T=S',S_2$, where $\#(\mathsf{S}')$ is the length of S'.

  If we are in the former case, then by induction hypothesis we have that $\langle S_1\rangle\xrightarrow{\{\!|\mu|\!\}}_{\{\!|l,\mathsf{S}_1,\mu|\!\}}\langle S_2\rangle$. Moreover by definition of the $\mathtt{nano}\pi$-calculus (we suppose that bond names never clash with free names) we have that $\mathtt{bn}(\{\!|\mu|\!\})\cap\mathrm{name}(\mathsf{S}')=\emptyset$. So one can apply the (lift) rule and obtain: $\langle\mathsf{S}\rangle=\langle\mathsf{S}_1\rangle|\langle\mathsf{S}'\rangle\xrightarrow{\{\!|\mu|\!\}}_{\{\!|l,\mathsf{S},\mu|\!\}}\langle S_2\rangle|\langle S'\rangle=\langle\mathsf{T}\rangle$.

  In the other case, the proof is achieved with the same arguments.

2. Suppose that $\mathsf{S}\xrightarrow{\rho}_{l,l'}\mathsf{T}$. This transition has been obtained by the (communications) rule so there exists S_1, S_2, T_1, T_2, ρ, $\imath$ and $\jmath$ such that:

   - $\mathsf{S}=\mathsf{S}_1,\mathsf{S}_2$ and $\mathsf{T}=\jmath(\mathsf{T}_1,\mathsf{T}_2)$,
   - $\mathsf{S}_1\xrightarrow{\rho_L,\imath}_l\mathsf{T}_1$,
   - and $\mathsf{S}_2\xrightarrow{\rho_R,\imath}_{l'-length(\mathsf{S}_1)}\mathsf{T}_2$

   where $\mathrm{dom}(\jmath)=\mathrm{name}(\mathsf{T}_1,\mathsf{T}_2)\setminus\mathrm{name}(\mathsf{S}_1,\mathsf{S}_2)$ and $\mathrm{ran}(\jmath)$ is the set of the least names not occurring in S.

   The remainder of the proof depends on ρ:

   - Suppose that ρ is a creation. By the first item of the theorem we have that:

– $\langle \mathsf{S}_1 \rangle \xrightarrow{\{\!|\rho_L,\imath|\!\}}_{\{\!|l,\rho_L,\imath,\mathsf{S}_1|\!\}} \langle \mathsf{T}_1 \rangle$,

– $\langle \mathsf{S}_2 \rangle \xrightarrow{\{\!|\rho_R,\imath|\!\}}_{\{\!|l'-length(\mathsf{S}_1),\rho_R,\imath,\mathsf{S}_2|\!\}} \langle \mathsf{T}_2 \rangle$,

and by the definition 4.3.2 there exists a, $\widetilde{u}$, $\widetilde{\lambda}$ and $\widetilde{v}$ such that $\{\!|\rho_L,\imath|\!\} = \overline{a}\,(\widetilde{u} : \widetilde{\lambda})$ and $\{\!|\rho_R,\imath|\!\} = a\,(\widetilde{v})$ (a is either ρ or a private name if respectively ρ as a bond in its left-hand or not), and such that $rate(a) = rate(\rho)$. So by the (communication) rule:

$$\{\!|\mathsf{S}|\!\} = \delta(\langle \mathsf{S}_1 \rangle | \langle \mathsf{S}_2 \rangle) \xrightarrow{\tau_{rate(\rho)}}_{\{\!|l,\rho_L,\imath,\mathsf{S}_1|\!\},\{\!|l'-length(\mathsf{S}_1),\rho_R,\imath,\mathsf{S}_2|\!\}+length(\langle \mathsf{S}_1 \rangle)} [(\delta + k(\widetilde{u}) : \widetilde{\lambda})(\langle \mathsf{T}_1 \rangle \{k(\widetilde{u})/\widetilde{u}\} | \langle \mathsf{T}_2 \rangle)\{k(\widetilde{u})/\widetilde{v}\}]_{\mathtt{GC}}$$

where $\mathrm{dom}(k) = \widetilde{u}$ and $\mathrm{ran}(k)$ are the least names not occurring in $\langle \mathsf{S}_1 \rangle | \langle \mathsf{S}_2 \rangle \setminus \{\widetilde{u} \bigcup \widetilde{v}\}$. Let $\{\!|\mathsf{T}|\!\}$ be written $\delta'(\langle \jmath(\mathsf{T}_1) \rangle | \langle \jmath(\mathsf{T}_2) \rangle)$. We obtain that:

$$\delta'(\langle \jmath(\mathsf{T}_1) \rangle | \langle \jmath(\mathsf{T}_2) \rangle) \overset{\circ}{\equiv}_{\alpha} [(\delta + k(\widetilde{u}) : \widetilde{\lambda})(\langle \mathsf{T}_1 \rangle \{k(\widetilde{u})/\widetilde{u}\} | \langle \mathsf{T}_2 \rangle)\{k(\widetilde{u})/\widetilde{v}\}]_{\mathtt{GC}}$$

by several arguments.

– First $\jmath$ maps the names created by ρ to the least names not occurring in S and k maps the names created by the synchronization to the least names not occurring in $\langle \mathsf{S}_1 \rangle | \langle \mathsf{S}_2 \rangle \setminus \{\widetilde{u} \bigcup \widetilde{v}\}$. These names might not be the same because the names in some $\{\!|\mathsf{S}''|\!\}$ are the names in S'' plus the names appearing as bound names of the inputs and bound outputs. However it suffices to rename these bound names to obtain that $\mathrm{ran}(\jmath) = \mathrm{ran}(k)$.

– Then since the names created by the synchronization are exactly the names created by ρ (as defined in definition 4.3.2) we obtain $\langle \jmath(\mathsf{T}_1) \rangle | \langle \jmath(\mathsf{T}_2) \rangle) \overset{\circ}{\equiv}_{\alpha} (\langle \mathsf{T}_1 \rangle \{k(\widetilde{u})/\widetilde{u}\} | \langle \mathsf{T}_2 \rangle)\{k(\widetilde{u})/\widetilde{v}\}$.

– Finally since ρ is a creation, by construction of the *set* functions, no name is removed after the synchronization. So the names declared in the encoding of T are exactly those declared in the encoding of S plus those created by the reaction: that is $\delta' = \delta + k(\widetilde{u}) : \widetilde{\lambda}$. And since no name is removed the operation $[.]_{\mathtt{GC}}$ has no effect.

Thus we obtain: $\{\!|\mathsf{S}|\!\} \xrightarrow{\tau_{rate(\rho)}}_{\{\!|l,\rho,\mathsf{S}_1|\!\},\{\!|l',\rho,\mathsf{S}|\!\}} \overset{\circ}{\equiv}_{\alpha} \{\!|\mathsf{T}|\!\}$.

- Suppose that ρ is a destruction. By the first item of the theorem we have that:

  – $\langle \mathsf{S}_1 \rangle \xrightarrow{\{\!|\rho_L,\imath|\!\}}_{\{\!|l,\rho_L,\imath,\mathsf{S}_1|\!\}} \langle \mathsf{T}_1 \rangle$,

  – $\langle \mathsf{S}_2 \rangle \xrightarrow{\{\!|\rho_R,\imath|\!\}}_{\{\!|l'-length(\mathsf{S}_1),\rho_R,\imath,\mathsf{S}_2|\!\}} \langle \mathsf{T}_2 \rangle$,

  and by the definition 4.3.2 there exists a, such that $\{\!|\rho_L,\imath|\!\} = \overline{a}\,()$ and $\{\!|\rho_R,\imath|\!\} = a\,()$, and such that $rate(a) = rate(\rho)$. So by the communication rule:

$$\{\!|\mathsf{S}|\!\} = \delta(\langle \mathsf{S}_1 \rangle | \langle \mathsf{S}_2 \rangle) \xrightarrow{\tau_{rate(\rho)}}_{\{\!|l,\rho_L,\imath,\mathsf{S}_1|\!\},\{\!|l'-length(\mathsf{S}_1),\rho_R,\imath,\mathsf{S}_2|\!\}+length(\langle \mathsf{S}_1 \rangle)} [\delta(\langle \mathsf{T}_1 \rangle | \langle \mathsf{T}_2 \rangle)]_{\mathtt{GC}}$$

Now since ρ is a destruction $\text{name}(\mathsf{T}_1|\mathsf{T}_2) \setminus \text{name}(\mathsf{S}) = \emptyset$ and so $\jmath$ is the empty renaming and $\langle \mathsf{T}_1 \rangle | \langle \mathsf{T}_2 \rangle = \langle \mathsf{T} \rangle$. Finally we have that $[\delta(\langle \mathsf{T}_1 \rangle | \langle \mathsf{T}_2 \rangle)]_{\mathtt{GC}} = \{\!|\mathsf{T}|\!\}$ because the update functions set_1 and set_2 removes the names corresponding to the deleted bond and because the $[.]_{\mathtt{GC}}$ operation remove the definition of these names.

- Suppose that ρ is an exchange. The proof of the case where it has the format $A[u](\sigma), B[v](\phi) \xrightarrow{\lambda} A[u'](\sigma), B[v'](\phi)$ is proved similarly to the case of the destruction: the arguments are the same except for the last point on the names which is eliminated since this reaction has no effects on names. Otherwise by the first item of the theorem we have that:
  - $\langle \mathsf{S}_1 \rangle \xrightarrow{\{\!|\rho_L, \imath|\!\}}_{\{\!|l, \rho_L, \imath, \mathsf{S}_1|\!\}} \langle \mathsf{T}_1 \rangle$,
  - $\langle \mathsf{S}_2 \rangle \xrightarrow{\{\!|\rho_R, \imath|\!\}}_{\{\!|l' - length(\mathsf{S}_1), \rho_R, \imath, \mathsf{S}_2|\!\}} \langle \mathsf{T}_2\{\mathtt{bn}(\{\!|\rho_\mathsf{R}, \imath|\!\})/\imath(\mathtt{EX}(\rho))\}\rangle$,

  and by the definition 4.3.2 there exists a, $\widetilde{u}$ and $\widetilde{v}$ such that $\{\!|\rho_L, \imath|\!\} = \overline{a}\,\widetilde{u}$ and $\{\!|\rho_R, \imath|\!\} = a\,(\widetilde{v})$, and such that $rate(a) = rate(\rho)$. So by the communication rule:

  $$\{\!|\mathsf{S}|\!\} = \delta(\langle \mathsf{S}_1 \rangle | \langle \mathsf{S}_2 \rangle) \xrightarrow{\tau_{rate(\rho)}}_{\{\!|l, \rho_L, \imath, \mathsf{S}_1|\!\}, \{\!|l' - length(\mathsf{S}_1), \rho_R, \imath, \mathsf{S}_2|\!\} + length(\langle \mathsf{S}_1 \rangle)} [\delta(\langle \mathsf{T}_1 \rangle | \langle \mathsf{T}_2\{\mathtt{bn}(\{\!|\rho_R, \imath|\!\})/\imath(\mathtt{EX}(\rho))\}\rangle\{\widetilde{u}/\widetilde{v}\})]_{\mathtt{GC}}$$

  Let $\{\!|\mathsf{T}|\!\}$ be written $\delta'(\langle \jmath(\mathsf{T}_1) \rangle | \langle \jmath(\mathsf{T}_2) \rangle)$. We obtain that $\delta'(\langle \jmath(\mathsf{T}_1) \rangle | \langle \jmath(\mathsf{T}_2) \rangle) = [\delta(\langle \mathsf{T}_1 \rangle | \langle \mathsf{T}_2\{\mathtt{bn}(\{\!|\rho_R, \imath|\!\})/\imath(\mathtt{EX}(\rho))\}\rangle\{\widetilde{u}/\widetilde{v}\})]_{\mathtt{GC}}$ by several arguments.
  - Since ρ is an exchange $\text{name}(T) = \text{name}(S)$, $\jmath$ is the empty renaming. Moreover no name is either created nor removed by the synchronization and so the operation $[.]_{\mathtt{GC}}$ has no effect and $\delta = \delta'$.
  - By the definition 4.3.2 we have that $\widetilde{v} = \mathtt{bn}(\{\!|\rho_R, \imath|\!\})$ and $\imath(\mathtt{EX}(\rho)) = \widetilde{u}$. So $\langle \jmath(\mathsf{T}_2) \rangle = \langle \mathsf{T}_2\{\mathtt{bn}(\{\!|\rho_R, \imath|\!\})/\imath(\mathtt{EX}(\rho))\}\rangle\{\widetilde{u}/\widetilde{v}\})$ and also $\langle \jmath(\mathsf{T}_1) \rangle = \langle \mathsf{T}_1 \rangle$.

  Thus we obtain: $\{\!|\mathsf{S}|\!\} \xrightarrow{\tau_{rate(\rho)}}_{\{\!|l, \rho, \mathsf{S}_1|\!\}, \{\!|l', \rho, \mathsf{S}|\!\}} \{\!|\mathsf{T}|\!\}$.

3. We prove the result by induction on the derivation tree of $\langle \mathsf{S} \rangle \xrightarrow{\mu}_{l.i} T$.
   - If the transition has been obtained by the (init) rule, then the solution S can be written $A[w](\tau)$ and by construction of the encoding (see definition 4.3.2) the i-th branch of $\langle \mathsf{S} \rangle = \widehat{A}(\{\!|w|\!\}_0, \{\!|\tau|\!\}_1, \{\!|\tau|\!\}_2)$ can be written $[\{\!|w|\!\}_0]_u[\{\!|\tau|\!\}_2]_\sigma\ \alpha_{\rho,side}.P_{\rho,side}$ where the matches are true and where ρ is a reaction of rate λ, $side \in \{L, R\}$ and $\mu = \alpha_{\rho,side}$ and such that:
     - either $\rho : A[u](\sigma) \xrightarrow{\lambda} \_ \in \mathcal{R}_R$, $side = R$ and $P_{\rho,side} = \langle \_ \rangle = 0$,
     - or $\rho : A[u](\sigma) \xrightarrow{\lambda} A[u'](\sigma') \in \mathcal{R}_L$, $side = L$ and $P_{\rho,side} = \widehat{A}(set_0(\{\!|w|\!\}_0, u'), set_1(\{\!|\tau|\!\}_1, \sigma, \sigma', \mathtt{bn}(\alpha_{\rho,side})), set_2(\{\!|\tau|\!\}_2, \sigma'))$,

– or $\rho : A[u](\sigma) \xrightarrow{\lambda} A[u'](\sigma'), \mathsf{R}' \in \mathcal{R}_R$, $side = R$ and $P_{\rho,side} = \widehat{A}(set_0(\{\!|w|\!\}_0, u'), set_1(\{\!|\tau|\!\}_1, \sigma, \sigma', \mathtt{bn}(\alpha_{\rho,side})), set_2(\{\!|\tau|\!\}_2, \sigma'))$, $\langle \mathsf{R}' \rangle$,

The proof of the two first cases are a subcase of the proof of the last case, so we suppose that we are in the last case. By Lemma 4.5.1 and since the matches are true we have that there exists v, $\imath$ and ν such that $w = u + v$ and $\tau = \imath \circ \sigma + \nu$. So by the (init) rule we obtain $A[u+v](\imath \circ \sigma + \nu) \overset{\rho_{side},\imath}{\longrightarrow}_1 A[u'+v](\imath \circ \sigma' + \nu), \langle \mathsf{R} \rangle$. The remainder of the proof depends on ρ_{side}.

– If ρ is an exchange then by Lemma 4.5.2 we have that $\langle \mathsf{S} \rangle = \widehat{A}(set_0(\{\!|u+v|\!\}_0, u'), set_1(\{\!|\imath \circ \sigma + \nu|\!\}_1, \sigma, \sigma', \mathtt{bn}(\alpha_{\rho,R})), set_2(\{\!|\imath \circ \sigma + \nu|\!\}_2, \sigma'), \langle \mathsf{R}' \rangle) = \langle A[u'+v](\imath \circ \sigma' + \nu)\{\mathtt{bn}(\alpha_{\rho,R})/\imath(\mathtt{EX}(\rho))\}, \mathsf{R}' \rangle$, since in $\mathtt{nano}\kappa$ the exchanged names are already passed to the products of the reaction at the step of the (init) rule, while in $\mathtt{nano}\pi$ it is only achieved by the communication during the step of the (communication) rule.

– Otherwise by Lemma 4.5.2 we have that $\langle \mathsf{S} \rangle = \widehat{A}(set_0(\{\!|u + v|\!\}_0, u'), set_1(\{\!|\imath \circ \sigma + \nu|\!\}_1, \sigma, \sigma', \mathtt{bn}(\alpha_{\rho,R})), set_2(\{\!|\imath \circ \sigma + \nu|\!\}_2, \sigma'), \langle \mathsf{R}' \rangle) = \langle A[u' + v](\imath \circ \sigma' + \nu), \mathsf{R}' \rangle$.

- If the transition has been obtained by the (lift) rule, then there exists S_1, S_2 and T' such that either $\langle \mathsf{S}_1 \rangle \xrightarrow{\mu}_{l.i} T'$, $\mathsf{S} = \mathsf{S}_1, \mathsf{S}_2$ and $\mathsf{T} = T', \langle \mathsf{S}_2 \rangle$ or $\langle \mathsf{S}_1 \rangle \xrightarrow{\mu}_{(l - length(\mathsf{S}_2)).i} T'$, $\mathsf{S} = \mathsf{S}_2, \mathsf{S}_1$ and $\mathsf{T} = \langle \mathsf{S}_2 \rangle, T'$.
If we are in the former case, then by the induction hypothesis there exists S' and μ' such that $\mathsf{S}_1 \xrightarrow{\mu'}_{l.i} \mathsf{S}'$, $\mu = \{\!|\mu'|\!\}$ and
  – either $\langle \mathsf{S}'\{\mu/\imath(\mathtt{GC}(\rho))\} \rangle = T'$ if $\mu' = \rho_R, \imath$ and if ρ is an exchange
  – or $\langle \mathsf{S}' \rangle = T'$ otherwise.

  By the lift rule we obtain $\mathsf{S} = \mathsf{S}_1, \mathsf{S}_2 \xrightarrow{\mu'}_{l.i} \mathsf{S}', \mathsf{S}_2$ (if ρ is a creation we have that $\mathrm{name}(\mathsf{S}') \setminus \mathrm{name}(\mathsf{S}_1) \bigcap \mathrm{name}(\mathsf{S}_2) = \emptyset$ because $\mathrm{name}(\mathsf{S}') \setminus \mathrm{name}(\mathsf{S}_1) \bigcap \mathrm{name}(\mathsf{S}_2) = \mathtt{bn}(\mu) \bigcap \mathrm{name}(\langle \mathsf{S}_2 \rangle)$ which is equal to $\emptyset$ by the (lift) rule). It suffices now to notice that:
  – either $\langle \mathsf{S}'\{\mu/\imath(\mathtt{GC}(\rho))\}, \mathsf{S}_2 \rangle = T$ if $\mu' = \rho_R, \imath$ and if ρ is an exchange,
  – or $\langle \mathsf{S}', \mathsf{S}_2 \rangle = T$ otherwise.

  The proof of the other case is achieved by the same arguments.

4. Suppose that $\{\!|\mathsf{S}|\!\} \xrightarrow{\tau_\lambda}_{l.i,l.j} T$. This transition has been obtained by the (communication) rule. So there exist δ, S_1, S_2, T_1, T_2 and μ such that $\{\!|\mathsf{S}|\!\} = \delta(\langle \mathsf{S}_1 \rangle, \langle \mathsf{S}_2 \rangle$ and $\langle \mathsf{S}_1 \rangle \xrightarrow{\mu}_{l.i} T_1$ and $\langle \mathsf{S}_2 \rangle \xrightarrow{\overline{\mu}}_{(l' - length(\mathsf{S}_1)).j} T_2$.
   By the third item of this theorem, there exist S'_1, S'_2 and μ' such that:

   - $\mathsf{S}_1 \xrightarrow{\mu'}_l \mathsf{S}'_1$, $\{\!|\mu'|\!\} = \mu$ and if $\mu' = \rho_R, \imath$ where ρ is an exchange then $\langle \mathsf{S}'_1\{\mathtt{bn}(\mu)/\imath(\mathtt{EX}(\rho))\} \rangle = T_1$ otherwise $\langle \mathsf{S}'_1 \rangle = T_1$,

- $\mathsf{S}_2 \xrightarrow{\overline{\mu'}}_{l-length(\mathsf{S}_1)} \mathsf{S}'_2$, $\{\!|\overline{\mu'}|\!\} = \overline{\mu}$ and if $\overline{\mu'} = \rho_R, \imath$ where ρ is an exchange then $\langle \mathsf{S}'_2\{\mathtt{bn}(\overline{\mu})/\imath(\mathtt{EX}(\rho))\}\rangle = T_2$ otherwise $\langle \mathsf{S}'_2\rangle = T_2$,

Let ρ be the rule of μ', we have that $\mathsf{S} = \mathsf{S}_1, \mathsf{S}_2 \xrightarrow{\rho}_{l,l'} \jmath(\mathsf{S}'_1, \mathsf{S}'_2)$ for some order-preserving injective renaming $\jmath$ mapping $\mathrm{name}(\mathsf{S}'_1, \mathsf{S}'_2) \setminus \mathrm{name}(\mathsf{S}_1, \mathsf{S}_2)$ to the least names not present in $\mathrm{name}(\mathsf{S}_1, \mathsf{S}_2)$. It remains to prove that $\{\!|\jmath(\mathsf{S}'_1, \mathsf{S}'_2)|\!\} = T$, this depends on ρ:

- If ρ is a creation, then we can suppose that the pair $(\mu, \overline{\mu})$ equals $(\overline{x}\,(\widetilde{u} : \widetilde{\lambda}), x\,(\widetilde{v}))$ for some channel x and some tuples of names $\widetilde{u}$ and $\widetilde{v}$. So since $\langle \mathsf{S}'_1\rangle = T_1$ and $\langle \mathsf{S}'_2\rangle = T_2$ we have $T = [(\delta + k(\widetilde{u}) : \widetilde{\lambda})(\langle \mathsf{S}'_1\rangle\{k(\widetilde{u})/\widetilde{u}\}|\langle \mathsf{S}'_2\rangle\{k(\widetilde{u})/\widetilde{v}\})]_{\mathtt{GC}}$ for some k mapping $\widetilde{u}$ to the least names not occurring in $\langle \mathsf{S}'_1\rangle|\langle \mathsf{S}'_2\rangle \setminus \{\widetilde{u} \bigcup \widetilde{v}\}$. Let us write $\{\!|\jmath(\mathsf{S}'_1, \mathsf{S}'_2)|\!\} = \delta'(\langle \jmath(\mathsf{S}'_1)\rangle|\langle \jmath(\mathsf{S}'_2)\rangle)$, we obtain that $\delta'(\langle \jmath(\mathsf{S}'_1)\rangle|\langle \jmath(\mathsf{S}'_2)\rangle) \overset{\circ}{\equiv}_\alpha [(\delta + k(\widetilde{u}) : \widetilde{\lambda})(\langle \mathsf{S}'_1\rangle\{k(\widetilde{u})/\widetilde{u}\}|\langle \mathsf{S}'_2\rangle\{k(\widetilde{u})/\widetilde{v}\})]_{\mathtt{GC}}$ by several arguments:
  - First $\jmath$ maps the names created by ρ to the least names not occurring in S and k maps the names created by the synchronization to the least names not occurring in $\langle \mathsf{S}'_1\rangle|\langle \mathsf{S}'_2\rangle \setminus \{\widetilde{u} \bigcup \widetilde{v}\}$. These names might not be the same because the names in some $\{\!|\mathsf{S}''|\!\}$ are the names in S'' plus the names appearing as bound names of the inputs and bonded outputs. However it suffices to rename these bonded names to obtain that $\mathrm{ran}(\jmath) = \mathrm{ran}(k)$.
  - Then since the names created by the synchronization are exactly the names created by ρ (as defined in definition 4.3.2) we obtain $\langle \jmath(\mathsf{S}'_1)\rangle|\langle \jmath(\mathsf{S}'_2)\rangle \overset{\circ}{\equiv}_\alpha \langle \mathsf{S}'_1\{k(\widetilde{u})/\widetilde{u}\}\rangle|\langle \mathsf{S}'_2\{k(\widetilde{u})/\widetilde{v}\}\rangle$.
  - Finally since ρ is a creation, by construction of the *set* functions, no name is removed after the synchronization. So the names declared in T are exactly those declared in the encoding of S plus those created by the reaction: that is $\delta' = \delta + k(\widetilde{u}) : \widetilde{\lambda}$. And since no name is removed the operation $[.]_{\mathtt{GC}}$ has no effect.
- If ρ is a destruction, then we can suppose that the pair $(\mu, \overline{\mu})$ equals $(\overline{x}\,(), x\,())$ for some channel x and some tuple of names $\widetilde{u}$. So since $\langle \mathsf{S}'_1\rangle = T_1$ and $\langle \mathsf{S}'_2\rangle = T_2$ we have $T = [\delta(\langle \mathsf{S}'_1\rangle|\langle \mathsf{S}'_2\rangle)]_{\mathtt{GC}}$. Let us write $\{\!|\jmath(\mathsf{S}'_1, \mathsf{S}'_2)|\!\} = \delta'(\langle \jmath(\mathsf{S}'_1)\rangle|\langle \jmath(\mathsf{S}'_2)\rangle)$. We obtain that $\delta'(\langle \jmath(\mathsf{S}'_1)\rangle|\langle \jmath(\mathsf{S}'_2)\rangle) = \langle(\delta)(T_1|T_2)\rangle_{\mathtt{GC}}$ because since ρ is a destruction $\jmath$ is the empty renaming, so $\langle \jmath(\mathsf{S}'_1)\rangle|\langle \jmath(\mathsf{S}'_2)\rangle = T_1|T_2$. Finally we have that $\delta'(\langle \jmath(\mathsf{S}'_1)\rangle|\langle \jmath(\mathsf{S}'_2)\rangle) = \langle(\delta)(T_1|T_2)\rangle_{\mathtt{GC}}$ because the update functions set_1 and set_2 remove the names corresponding to the deleted bond and because the $[.]_{\mathtt{GC}}$ operation removes the definition of these names.
- If ρ is an exchange then we can suppose that the pair $(\mu, \overline{\mu})$ equals $(\overline{x}\,\widetilde{u}, x\,(\widetilde{v}))$ for some channel x and some tuples of names $\widetilde{u}$ and $\widetilde{v}$. So since $\langle \mathsf{S}'_1\rangle = T_1$ and $\langle \mathsf{S}'_2\{\mathtt{bn}(\overline{\mu})/\imath(\mathtt{EX}(\rho))\}\rangle = T_2$ we have $T = [\delta(\langle \mathsf{S}'_1\rangle|\langle \mathsf{S}'_2\{\mathtt{bn}(\overline{\mu})/\imath(\mathtt{EX}(\rho))\}\rangle\{\widetilde{u}/\widetilde{v}\})]_{\mathtt{GC}}$. Let us write $\{\!|\jmath(\mathsf{S}'_1, \mathsf{S}'_2)|\!\} = \delta'(\langle \jmath(\mathsf{S}'_1)\rangle|\langle \jmath(\mathsf{S}'_2)\rangle)$. We obtain that:

$$\delta'(\langle \jmath(\mathsf{S}'_1)\rangle | \langle \jmath(\mathsf{S}'_2)\rangle) = [\delta(\langle \mathsf{S}'_1\rangle | \langle \mathsf{S}'_2\{\mathtt{bn}(\overline{\mu})/\imath(\mathtt{EX}(\rho))\}\rangle\{\widetilde{u}/\widetilde{v}\})]_{\mathtt{GC}}$$

by several arguments.

- Since ρ is an exchange $\jmath$ is the empty renaming. Moreover no name is either created nor removed by the synchronization and so the operation $[.]_{\mathtt{GC}}$ has no effect and $\delta = \delta'$.
- By the definition 4.3.2 we have that $\widetilde{v} = \mathtt{bn}(\overline{\mu})$ and $\imath(\mathtt{EX}(\rho)) = \widetilde{u}$. So $\langle \jmath(\mathsf{S}'_1)\rangle = \langle \mathsf{S}'_1\rangle$ and $\langle \jmath(\mathsf{S}'_2)\rangle = \langle \mathsf{S}'_2\{\mathtt{bn}(\overline{\mu})/\imath(\mathtt{EX}(\rho))\}\rangle\{\widetilde{u}/\widetilde{v}\}$.

□

4.5.3 Correctness of the encoding with respect to the collective transition relation.

Theorem 4.5.3 *Given any pair of languages $\mathfrak{C}_1$ and $\mathfrak{C}_2$, let L_1 and I_1, and L_2 and I_2 respectively be the label sets (i.e. the symbols on top of the transitions) and the tag sets (i.e. the subscripts of the transitions) of their basic transition relation and $\equiv_1$ and $\equiv_2$ their structural equivalences. With an abuse of notation we denote respectively by $\longrightarrow$ and $\longmapsto$ the basic and collective transition relation of both calculi.*

For any encoding $(\!|.|\!)$ from $\mathfrak{C}_1$ to $\mathfrak{C}_2$, if there exist two injective maps σ from $\mathfrak{C}_1 \times L_1 \times I_1$ to L_2 and τ from $\mathfrak{C}_1 \times L_1 \times I_1$ to I_2, such that:

- $rate(\sigma(P,\rho,i)) = rate(\rho)$
- $P \xrightarrow{\rho}_i Q$ *implies* $(\!|P|\!) \overset{\sigma(P,\rho,i)}{\longrightarrow}_{\tau(P,\rho,i)} \overset{\circ}{\equiv}_\alpha (\!|Q|\!)$
- $(\!|P|\!) \xrightarrow{\rho}_i Q$ *implies that there exist* Q', ρ' *and* i' *such that* $(\!|Q'|\!) \overset{\circ}{\equiv}_\alpha Q$, $P \xrightarrow{\rho'}_{i'} Q'$, $\sigma(P,\rho',i') = \rho$ *and* $\tau(P,\rho',i') = i$
- $P \equiv_1 Q \Leftrightarrow (\!|P|\!) \equiv_2 (\!|Q|\!)$

then $P \overset{\lambda}{\longmapsto} Q$ if and only if there exists Q' such that $(\!|P|\!) \overset{\lambda}{\longmapsto} Q'$ and $Q' \equiv (\!|Q|\!)$.

Before proving Theorem 4.5.3, we state and prove the following lemma:

Lemma 4.5.3 *The hypotheses of Theorem 4.5.3 imply that:*

$$\sum_{((\mu,l),R)\in[next(P)]_Q} rate(\mu) = \sum_{((\sigma(P,\mu,l),\tau(P,\mu,l)),(\!|R|\!))\in[next((\!|P|\!))]_{(\!|Q|\!)}} rate(\sigma(P,\mu,l))$$

Proof The second and third items of the hypothesis imply that $((\mu,l),R) \in [next(P)]_Q$ if and only if $((\sigma(P,\mu,l),\tau(P,\mu,l)),(\!|R|\!)) \in [next((\!|P|\!))]_{(\!|Q|\!)}$. Therefore $\sum_{((\mu,l),R)\in[next(P)]_Q} rate(\mu) = \sum_{((\sigma(P,\mu,l),\tau(P,\mu,l)),(\!|R|\!))\in[next((\!|P|\!))]_{(\!|Q|\!)}} rate(\mu)$. The result is now obtain by using the first item of the hypothesis.

□

We are now ready to prove Theorem 4.5.3.

Proof (of Theorem 4.5.3)

- Proof of $P \stackrel{\lambda}{\longmapsto} Q \Rightarrow (\!|P|\!) \stackrel{\lambda}{\longmapsto} (\!|Q|\!)$.

  By construction of the collective transition relation, there exists a term $Q' \in \mathfrak{C}_1$, an identifier $l \in I_1$ and a label $\mu \in L_1$ such that $P \stackrel{\mu}{\longrightarrow}_l Q'$, $Q \equiv_1 Q'$ and moreover $\lambda = \sum_{((\mu',l'),R) \in [next(P)]_{Q'}} rate_1(\mu')$. By Lemma 4.5.3, $\lambda = \sum_{((\sigma(P,\mu',l'),\tau(P,\mu',l')),(\![R]\!)) \in [next((\![P]\!))]_{(\![Q]\!)'}} rate_2(P, \sigma(\mu', l'))$. So $(\![P]\!) \stackrel{\lambda}{\longmapsto} can([next((\![P]\!))]_{(\![Q']\!)})$ by construction of the collective transition relation.

  By definition of $can(.)$, $can([next((\![P]\!))]_{(\![Q']\!)}) \equiv_2 (\![Q']\!)$. Since $Q \equiv_1 Q'$ and by the last item of the hypothesis, $(\![Q']\!) \equiv_2 (\![Q]\!)$. So $can([next((\![P]\!))]_{(\![Q']\!)}) \equiv_2 (\![Q]\!)$ and so $(\![P]\!) \stackrel{\lambda}{\longmapsto} \stackrel{\circ}{\equiv}_\alpha (\![Q]\!)$.

- Proof of $(\![P]\!) \stackrel{\lambda}{\longmapsto} (\!|Q|\!) \Rightarrow P \stackrel{\lambda}{\longmapsto} Q$.

  By construction of the collective transition relation, there exists a label $\mu' \in L_2$, an identifier $l' \in I_2$ and a term $R' \in \mathfrak{C}_2$ such that $(\![P]\!) \stackrel{\mu'}{\longrightarrow}_{l'} R'$, $R' \equiv_2 (\![Q]\!)$ and moreover $\lambda = \sum_{((\mu,l),T) \in [next((\![P]\!))]_{R'}} rate_2(\mu)$. By the third item of the hypothesis, this sum can be rewritten into

  $$\sum_{((\sigma(P,\mu',l'),\tau(P,\mu',l')),(\![T']\!)) \in [next((\![P]\!))]_{R'}} rate_2(\sigma(P, \mu', l'))$$

  By lemma 4.5.3 and since $(\![Q]\!) \equiv R'$ implies $[next(P)]_{(\![Q]\!)} = [next(P)]_{R'}$, we obtain $\lambda = \sum_{((\mu',l'),T') \in [next(P)]_Q} rate_2(\mu')$. So $P \stackrel{\lambda}{\longmapsto} can([next(P)]_Q)$ by construction of the collective transition.

  By definition of $can(.)$, $can([next_\lambda(P)]_Q) \equiv_1 Q$. Conclusion $P \stackrel{\lambda}{\longmapsto} Q$.

□

We can now state our final correctness theorem.

Theorem 4.5.4 $\mathsf{S} \stackrel{\lambda}{\longmapsto} \mathsf{T}$ *in* nanoκ *if and only if there exists* P *such that* $\{\!|\, (\!|\mathsf{S}|\!) \,|\!\} \stackrel{\lambda}{\longmapsto} P$ *and* $P \equiv \{\!|\, (\!|\mathsf{T}|\!) \,|\!\}$ *in* nanoπ.

Proof We prove the result by applying Theorem 4.5.3 on the two encodings $(\!|.|\!)$ and $\{\!|.|\!\}$. For $(\!|.|\!)$, one defines $\sigma(\mathsf{S}, \rho, (l, l')) = \rho$ and $\tau(\mathsf{S}, \rho, (l, l')) = (l, l')$. For $\{\!|.|\!\}$, one defines $\sigma(\mathsf{S}, \rho, (l, l')) = \tau_\lambda$ where λ is the rate of ρ and $\tau(\mathsf{S}, \rho, (l, l')) = (l.i, l'.i)$ where i is the index of the reaction ρ. Theorems 4.5.1 and 4.5.2 imply that these definitions fit the first three hypotheses of Theorem 4.5.3. Moreover, it is easy to see that for any nanoκ solutions S and T $\mathsf{S} \equiv \mathsf{T} \Leftrightarrow (\!|\mathsf{S}|\!) \equiv (\!|\mathsf{T}|\!)$ and for any nanoκ solutions with gangs S and T, $\mathsf{S} \equiv \mathsf{T} \Leftrightarrow \{\!|\mathsf{S}|\!\} \equiv \{\!|\mathsf{T}|\!\}$. □

4.6 Conclusion and Future work.

Our current interests are mainly in simulators and analysis tools for nanoπ-calculus. In fact, this contribution allows us to simulate nanoκ systems. However, the same encoding also makes it possible to model-check nanoκ formalizations in the PRISM platform [29], since it supports verifications of probabilistic and stochastic extensions of π-calculus [38]. More precisely, it should be possible to wire our encoding from the nanoκ calculus to nanoπ with the implementation in [38]. There are two questions of concern. Firstly, our encoding uses polyadic communications, which is not considered in [38]. However this should be one of the next extensions of this work. The second issue is more problematic. A relevant constraint for the efficiency of the encoding in [38] is the absence of name creations within agent definition. This is not the case for our encoding, because agents may perform bonded outputs. Yet, in nanoκ subsystems where the creation of new molecules is finite, the number of names used at every stage of the computation is finite. So, a clever algorithm might compute this number statically (an over-approximation is $k \times h$, where k is the maximal number of molecules and h is the maximal length of the arguments of an agent) and use a garbage-collection mechanism to recycle names. This should allow the static allocation of variables in the PRISM language to handle all the private names.

4.7 Related work.

In [66], it has been shown that systems of molecular interactions with explicit bonds might be represented and simulated using the stochastic π-calculus. Our encoding corroborates this result since the nanoπ-calculus is a subset of the stochastic π-calculus. We remark that the example provided in [66] and, we believe, the descriptions done in this approach, can easily be rewritten in nanoπ-calculus and even in a sub-calculus of it, since our encodings does not use its full power.

Encodings from the full κ-calculus to nanoκ calculus, or to π-calculus are presented in [26] and [23]. Yet, they only preserve non-stochastic semantics and would hardly preserve the stochastic semantics since they do not preserve the granularity. In facts, encodings preserving the stochastic semantics do not exist, due to the negative results of [51].

In [14], Cardelli has encoded chemical systems into process algebra and back preserving both the stochastic and the ODE semantics. Our encoding extends these encodings because the CGF process algebra used in [14] is a subset of the nanoπ-calculus and because the nanoκ calculus extends the language of chemical reactions of [14] with explicit bonds between molecules and with internal states. However, our results are weaker than those in [14], since we only assert the correctness of the encoding with respect to the stochastic semantics.

Another stochastic process calculus that has been used also for the modeling of biochemical systems is PEPA [43]. For instance, in [12], PEPA has been exploited to examine the influence of the Raf Kinase Inhibitor Protein (RKIP)

on the Extracellular signal Regulated Kinase (ERK) signaling pathway. Also in that paper, as in the present one, a reagent-centric view and a pathway-centric (process-centric in our terminology) view are studied. Our analysis of the two approaches is different for two main reasons. First of all, in the PEPA-based approach one process is used to represent the concentration of one species while we follow the Cardelli's approach considering one process for each molecule. In fact, we have found this approach appropriate for a compositional model of discrete state systems (in which we count the number of molecules instead of considering their concentrations). The second difference is that in [12] only finitely many different species are considered, thus the translation from the reagent-centric to the pathway-centric views can be obtained using an intermediate matrix representation that quantifies the impact of each reaction on each reagent in a manner analogous to the stoichiometry matrix of the chemical reactions. We cannot exploit this approach as we do not impose any bound on the number of different complexes that can be produced in a `nano`κ calculus system. More recently, the PEPA approach has been also extended with a reaction-centric model called Bio-PEPA [21]. Also in this case, one process is used to represent the concentration of one species.

Chapter 5

Comparing the chemical master equation and the backward stochastic bisimulation.

5.1 Introduction.

In this Chapter we study the relationship between the chemical and the computer science semantics. An important chemical semantics is the CME. Important semantics of the process algebra are the various equivalences. We investigate an equivalence obtained by reasoning on the CME. Interestingly, it turns out that this equivalence corresponds exactly to the an equivalence already studied in computer science [59, 69]: the *backward stochastic bisimulation*.

The Chapter is organized as follows. In Section 5.2 we present the CME and motivate the CME-equivalence, in Section 5.3 we present the backward stochastic bisimulation and in Section 5.4 we state and prove the correspondence between the two semantics. We also provide some examples of encodings preserving the master equation equivalence. The Chapter is closed by a conclusion in Section 5.5 and a discussion on related works in Section 5.6.

5.2 Chemical master equation and chemical equivalences.

5.2.1 The chemical master equation.

Given a CTMC (S, μ, e_0) we note $Exit(e)$ the exit rate of the state e defined by $\sum_{f \in \mathcal{S}} \mu(e, f)$. The definition of the CME follows:

Definition 5.2.1 *Given a CTMC (S, μ, e_0), let $P(e, \tau)$ be the probability of the system being in state e at time τ knowing that the system was in state e_0 at time 0. The CME associated with the CTMC is defined by the following set of equations, for all $e \in S$:*

$$\begin{aligned} \frac{\delta P(e,\tau)}{\delta \tau} &= \sum_{f \in S} [\mu(f,e).P(f,\tau) - \mu(e,f).P(e,\tau)] \\ \textit{or equivalently} & \\ &= \sum_{f \in S} [\mu(f,e).P(f,\tau)] - Exit(e).P(e,\tau) \end{aligned}$$

The CME provides a full description of the probabilistic behavior of chemical systems by the dynamics of the discrete populations of molecules. It is a differential equation on the probabilities $P(e, \tau)$ that can be derived from the Chapman-Kolmogorov equation [71]. Since it is usually very hard to solve, techniques such as the Gillespie's algorithm have been developed to simulate its solution [37].

The term $\mu(f, e).P(f, t)$ contributes positively to the derivative since it corresponds to the entering of e from f at time τ, and conversely $\mu(e, f).P(e, \tau)$ contributes negatively since it corresponds to the exiting of e to f at time τ.

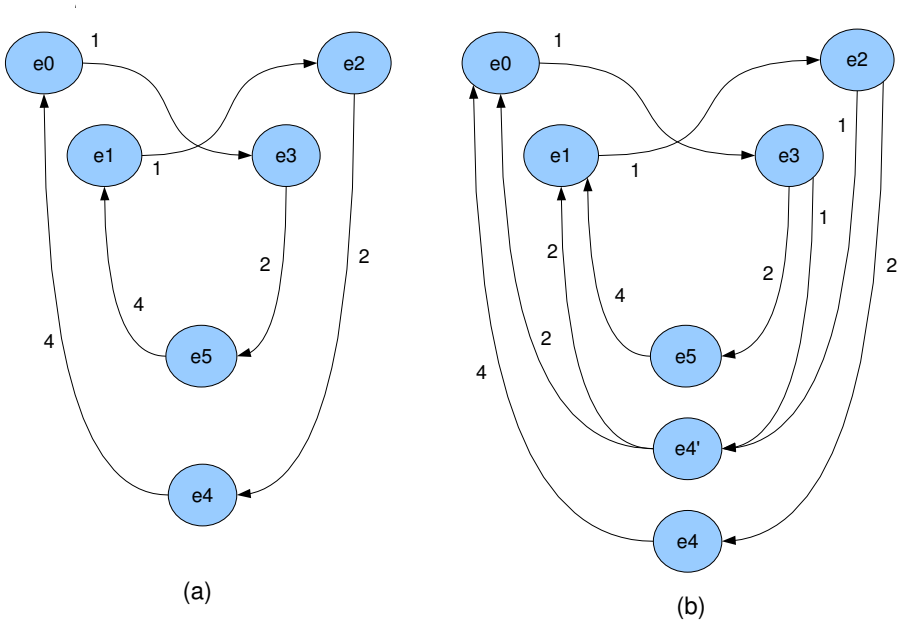


Figure 5.1: Two CTMCs.

Notation For the sake of the readability we often write e instead of $P(e,\tau)$ and $\dot{e}$ instead of $\frac{\delta P(e,\tau)}{\delta \tau}$.

5.2.2 Chemical equivalences.

We now motivate our definition of an equivalence based on the `CME` *from a purely chemistry-related point of view*. We first obtain the notion of `CME`-permutability (definition 5.2.2) which is not satisfying and therefore we carry on with the notion of `CME`-substitution (definition 5.2.3) which is going to be our notion of `CME`-equivalence.

We want to consider as equivalent two states of a CTMC which have similar equation in the `CME` or similar role inside the `CME`. If the notion of similarity is defined appropriately, this should permit us to identify states with have the same stochastic behavior. Consider for instance the CTMC of Figure 5.1 (a).

The CME is the following set of equations:

$$\begin{aligned}
\dot{e_0} &= 4 * e_4 - 1 * e_0 \\
\dot{e_1} &= 4 * e_5 - 1 * e_1 \\
\dot{e_2} &= 1 * e_1 - 2 * e_2 \\
\dot{e_3} &= 1 * e_0 - 2 * e_3 \\
\dot{e_4} &= 2 * e_2 - 4 * e_4 \\
\dot{e_5} &= 2 * e_3 - 4 * e_5
\end{aligned}$$

The polynomials $4 * e_4 - 1 * e_0$ and $4 * e_5 - 1 * e_1$ of the first two equations are not exactly the same, however they are similar enough to lead to similar behavior. Indeed they only differ on variables e_4 and e_5, and e_0 and e_1, which correspond to states with symmetric roles in the CME. So permuting e_0 with e_1, e_2 with e_3, and e_4 with e_5, leaves the set of equations unchanged and so the stochastic behaviors. We formalize this idea in the following definition.

Definition 5.2.2 *A* CME*-permutation is a permutation σ on the set of states such that whenever $f = \sigma(e)$, if $\dot{e} = \sum_g [\alpha_g.g] - \alpha.e$ then $\dot{f} = \sum_g [\alpha_g.\sigma(g)] - \alpha.f$.*

Two states e and e' are permutable in a CME *if there exists a* CME*-permutation σ such that $\sigma(e) = e'$.*

In words, by applying σ on the variables of the equation of e, one obtains the equation of $\sigma(e)$. So the CME is stable by σ. In Figure 5.1(a), the pair of states e_0 and e_1, e_2 and e_3 and e_4 and e_5 are permutable.

This notion should however be relaxed. Consider the example of Figure 5.1(b). The pair of states e_0 and e_1, e_2 and e_3, and e_4 and e_5 are permutable but the state e'_4 is not permutable with any other state. However we would like to say that it has an equation similar to the one of e_4 and e_5. Indeed the corresponding equations are:

$$\begin{aligned}
\dot{e'_4} &= 1 * e_2 + 1 * e_3 - 4 * e'_4 \\
\dot{e_4} &= 2 * e_2 - 4 * e_4 \\
\dot{e_5} &= 2 * e_3 - 4 * e_5
\end{aligned}$$

Suppose that e_2 and e_3 are equivalent, that is they have the same stochastic behavior, then they contribute similarly to any equation in the CME, and one can replaced one with the other. Thus the terms $1 * e_2 + 1 * e_3$, $2 * e_2$ and $2 * e_3$ should be equivalent. We formalize this in the following definition.

Definition 5.2.3 *A* CME*-substitution ϕ, is an application whose domain is the set of states, such that whenever $\phi(e) = \phi(f)$, if $\dot{e} = \sum_g [\alpha_g.g] - \alpha.e$ and*

$\dot{f} = \sum_h [\beta_h.h] - \beta.f$, *then* $\sum_g [\alpha_g.\phi(g)] - \alpha.\phi(e)$ *and* $\sum_h [\beta_h.\phi(h)] - \beta.\phi(f)$ *are the same polynomials (the terms* $\phi(g)$, $\phi(e)$, $\phi(h)$ *and* $\phi(f)$ *are the variables of the polynomials and the terms* α_g, α, β_h *and* β *are the coefficients of the polynomials).*

Two states e *and* e' *are* CME*-equivalent in a* CME *if there exists a* CME*-substitution* ϕ *such that* $\phi(e) = \phi(e')$. *We then write* $e \sim_{\texttt{CME}} e'$.

In words, if $\phi(e) = \phi(f)$ then by applying ϕ on the equations of e and f, one obtains the same equation. In Figure 5.1, $e_0 \sim_{\texttt{CME}} e_1$, $e_2 \sim_{\texttt{CME}} e_3$ and $e_4 \sim_{\texttt{CME}} e_5 \sim_{\texttt{CME}} e'_4$. The notion of CME-permutability is weaker than the notion of CME-equivalence.

Proposition 5.2.1 *If two states of a CTMC are* CME*-permutable then they are* CME*-equivalent.*

Proof. Given a CME-permutation σ choose a representative for each equivalence class of σ and define the substitution ϕ such that $\phi(e)$ is equals to the representative of the class to which e belongs. Then the properties of σ imply that ϕ is a CME-substitution.

The following proposition states the soundness of our definition:

Proposition 5.2.2 *For all states* e *and* e', *if* $P(e,0) = P(e',0)$ *and* $e \sim_{\texttt{CME}} e'$ *then at every instant* τ *one has* $P(e,\tau) = P(e',\tau)$.

Proof. The results could be proved directly but it is easier to obtain from the correspondence with the backward stochastic bisimulation: it is an immediate corollary of Proposition 5.3.1 and Theorem 5.4.1.

5.3 The backward stochastic bisimulation.

The notion of forward bisimulation has been widely studied (see [3, 9, 43] among others). On the contrary the notion of backward stochastic bisimulation has already been studied but quite rarely [59, 69]. While the traditional notion of forward bisimulation concerns the *outgoing* transitions of a process, the backward bisimulation concerns the *ingoing* transitions and the *exit rate*.

Definition 5.3.1 *An equivalence relation* $\mathfrak{R}$ *on the set of states of a CTMC is a backward stochastic bisimulation if and only if whenever* $e\,\mathfrak{R}\,f$ *:*

- *for any* $\mathfrak{R}$*-equivalence class* C, $\mu(C,e) = \mu(C,f)$
- $Exit(e) = Exit(f)$

Two states e *and* f *are backward bisimilar if there exists a backward bisimulation* $\mathfrak{R}$ *such that* $e\mathfrak{R}f$. *Then we write* $e \sim_b f$.

In words when two states are equivalent then the cumulative rate from every equivalence class of the bisimulation to these states are the same, and their exit rate are the same. As usually one can prove that the arbitrary union of bisimulations is again a bisimulation. The union of all bisimulation is called bisimilarity.

In the traditional presentation of bisimulation, transitions are labeled with actions. In this setting we can restrict ourselves to τ-transitions and use only rates as labels since we are concerned only with the global state of the system: no interaction with the environment is to be expected. However it could interesting to try to find a chemical counterpart for the labeled version of the bisimulation.

As it was already pointed out in [69] it is worth to notice that two backward bisimilar states have the same probability.

Proposition 5.3.1 *For all states e and e', if $P(e,\tau) = P(e',\tau)$ and $e \sim_b e'$ then at every instant τ one has $P(e,\tau) = P(e',\tau)$.*

The results holds because the theory of lumpability [11] has shown that it holds for exact lumping and because the backward stochastic bisimulation corresponds to the exact lumping (in the same way that forward stochastic bisimulation corresponds to ordinary lumping).

5.4 The correspondence between the two semantics.

We are now ready to prove the correspondence between the two semantics: the following theorem states that the chemical semantics, the CME-equivalence, is equal to the computer science semantics, the backward stochastic bisimulation.

Theorem 5.4.1 *Two states of a CTMC are* CME*-equivalent if and only if they are backward stochastic bisimilar:*

$$e \sim_{\texttt{CME}} f \Leftrightarrow e \sim_b f$$

We call this equivalence the master equation equivalence *and note it $\sim$.*

Proof of the "if" direction. Given a backward bisimulation $\mathfrak{R}$, choose a representative for each equivalence class of $\mathfrak{R}$. Then define the substitution ϕ such that $\phi(e)$ is equals to the representative of the class to which e belongs. The properties of $\mathfrak{R}$ imply that ϕ is a CME-substitution.

Proof of the "only if" direction. Given a CME-substitution ϕ, define the relation $\mathfrak{R}$ by $s\mathfrak{R}t$ if and only if $\phi(e) = \phi(f)$. The properties of ϕ imply that $\mathfrak{R}$ is a backward bisimulation.

Examples. We carry on with an example of stochastic encoding which preserves the master equation equivalence.

In [14] L.Cardelli provided encodings from a dialect of CCS, the CGF-calculus, and formal system of chemical reaction and vice-versa. The encodings preserve both the discrete and continuous semantics, that is the ordinary differential equation semantics and the stochastic semantics. In [18] he provided a master equation for process algebra and in particular showed that his encodings preserve the master equation in the sense that a system of chemical reaction or a process algebra term and their encodings have the same set of equations. This implies that these encodings preserve our notion of master equation equivalence.

In the previous Chapter we have presented an encoding from the `nano`κ calculus to the `nano`π-calculus. It satisfies the following correctness property: $S \overset{\lambda}{\longmapsto} T \Leftrightarrow [\![S]\!] \overset{\lambda}{\longmapsto} \equiv [\![T]\!]$, where S and T are `nano`κ solutions, $[\![S]\!]$ and $[\![T]\!]$ are their encodings and λ is a stochastic rate. Now consider a given `nano`κ calculus system and call $\mathcal{C} = (\mathcal{S}, \mu, s_0)$ the CTMC obtained by computing its stochastic collective transition relation and downgrading it, and call $[\![\mathcal{C}]\!] = ([\![\mathcal{S}]\!], [\![\mu]\!], [\![s_0]\!])$ the CTMC whose states are the encodings of the states of $\mathcal{S}$ and whose stochastic transition relation is obtained as the downgrading of the stochastic collective transition relation of the `nano`π-calculus on these terms. By the correctness property of the encoding we have that each state $e \in \mathcal{S}$ and its encoding $[\![e]\!]$ have the same ingoing and outgoing transitions and so are equivalent by the master equation equivalence.

5.5 Conclusion.

We have presented the master equation equivalence that is motivated by the `CME` and proved that it corresponds to the backward bisimulation. This establishes a bridge between chemistry and process algebra at the semantics level.

It is worth noticing that our interest for the ingoing transitions raise a specific problem:

> Given a term Q, is the set of terms P that reduce to Q decidable ? Is it finitely computable ?

The corresponding question for the traditional equivalences concerns the computability and decidability of the outgoing transitions, since the traditional equivalences rely only on the outgoing transitions. This question is not straightforward and has already been studied in [72]: the main difficulties are probably the decidability of the structural congruence, the possible infinite branching introduced by the name creation and the unguarded recursion.

This indicates that the decidability and computability of the ingoing transitions is not straightforward and that we should take care of it in order to assess the relevance of the master equation equivalence. We can already provide some insights.

In the case of rule-based languages such has the `nano`κ calculus, the full κ-calculus or the Bigraphs [56, 50], there are usually a finite number of rules

and each of them can be applied in a finite number of ways to a given a term. So the outgoing transitions are finitely computable. If moreover the rules are reversible, that is, if whenever $P \longrightarrow Q$ is a valid rule then so is $Q \longrightarrow P$, then the ingoing transitions can be computed similarly to the outgoing ones. In Bigraphs rules are always reversible and also in $\texttt{nano}\kappa$ if we prevent the creation and destruction of molecules.

In the case of the $\texttt{nano}\pi$ some issues are also worth noting. For a given set of agent definitions, we can compute all the pairs of choice branches that could synchronize for some proper instantiations of the parameters. Then we can compute the set of the continuations corresponding to these synchronizations. Finally it suffices to look for these continuations that are subterms of the process Q.

However the main problem with these derivations of the ingoing transitions is that they also compute the states that lead to Q but that would not be reachable by forward transitions. And the interest of such states is arguable since they do not appear in any execution of the system. Therefore we consider looking for some constraints that ensure that states are backward reachable if and only if they are forward reachable.

Future works: toward a biochemical metric. In the stochastic setting, equivalences require that the stochastic rate of the transitions matched by the equivalence have to be exactly the same. This requirement is often too strong in practice because of the sensitivity it imposes on these numbers. It is particularly true in the case of biochemical systems where the data are measured with a finite precision. Metrics have been introduced to cope with this problem in the probabilistic setting. A metric is a distance between processes such that two processes are probabilistically bisimilar whenever their distance is 0. Moreover this distance should be continuous in the probabilities of the processes. There also exist algorithms to compute the distance between two processes up to a given error [32, 33].

We wish to build a metric on top of the master equation equivalence. The establishing of such a metric would have many applications. A first one would be to reduce the size of the state space of a model: given an error ϵ one can aggregate groups of states supposing that their distance is less than ϵ. This seems particularly relevant in the case of the approximate model-checking [40]. Another application is the sensitivity analysis: the sensitivity of a process P to a given parameter could be studied by computing the distance between P and P where the parameter has been slightly modified.

The standard approach to the building of a metric consists of three steps. First, a logic characterizing the considered bisimulation is defined: that is, a logic such that two processes are bisimilar if and only if they satisfy the same set of formulas. Then a satisfaction degree of a formula is defined. And finally, the distance between two processes is the greatest difference of satisfaction degree of the processes on all the formulas. We can be optimistic that such a logic can be defined in the case of the backward stochastic bisimulation because such

a logic exists for both stochastic bisimulation [34] and classical backward bisimulation [59]. The definition of the satisfaction degree might be more difficult but the example of the probabilistic case seems a good trail to follow.

5.6 Related Works.

Connection between the process algebra field and the chemical master equivalence is not unheard of. In [18] Cardelli provided a notion of master equation for process algebra and proved that its encodings between systems of chemical reactions and process algebra [14] preserve exactly the notion of master equation. We believe that his approach and our are complementary: he extends the notion of `CME` to the field of process algebra while we provide a correspondence between the `CME` and an equivalence already studied in computer science.

Chapter 6

Expressiveness of stochastic and probabilistic π-calculi.

6.1 Introduction.

In this Chapter we present an unexpected consequence of our study of nano-devices on the expressiveness of the π-calculus.

The stochastic settings. We focus on one of the key mechanisms of Concurrency: the choice operator. The process $P_1 + \ldots + P_n$ can "choose" to evolve as one of the P_i. A choice is guarded if each branch is prefixed by an action, that is each P_i in the above choice is written $\alpha_i.P'_i$. The choices are traditionally classified according to the possible guards:

- *input-guarded choice*: the guards can only be input actions,
- *output-guarded choice*: the guards can only be output actions,
- *separate choice*: a choice can contain input or output guards, but not both,
- *mixed choice*: a choice can contain both input and output guards,

In the classical settings, it has been proved that the asynchronous π-calculus (without choice and with asynchronous outputs) can encode the separate choice [57, 58], but not the mixed choice [60]. We investigate whether a similar gap holds in the stochastic settings. It turns out that such a gap exists between the separate and mixed stochastic choices when both calculi have or do not have infinite rates. Moreover we also show that, under a reasonable assumption, the addition of infinite rates to the separate choice permits us to encode the mixed choice without infinite rates.

In order to establish the relative expressive power of two languages our criterion is the existence or non-existence of "admissible" encodings between them. Our notion of admissible encoding is mainly inspired from the *operational correspondence* [61]. Intuitively we ask that a term and its encoding exhibit the same finite rate transitions. In the case of a separation result, a standard approach consists of exhibiting a problem that can be solved using one language but not using the other, and such that a "reasonable" encoding preserves the solutions of the problem. The existence of such a problem ensures the absence of a "reasonable" encoding. This was for instance the approach adopted in [60]: the "leader election" problem can be solved with processes having mixed choices, but not with processes having only the separate choice. Interestingly, our separation results are proved using the following problem:

Given a fragment of the stochastic π-calculus, is there an encoding from the $\mathtt{nano}\kappa$ *calculus to this language?*

Since the separate choice is not sufficient to perform the encoding that we have presented in Chapter 4, but the mixed choice is, the above problem permits us to establish our separation results between the separate choice and the mixed one in the stochastic settings.

A hierarchy of rates. As mentioned above the separate choice with infinite rate can encode the mixed choice without infinite rate, but no with infinite rate. Supposing a higher order of infinity in the rates, could the separate choice equipped with these "bigger" infinite rate, encode the mixed choice with normal infinite rate? To investigate this question in the Section 6.5 we introduce the multi-scale π-calculus where rates can be of different orders of magnitude. That is instead of having just finite rates belonging to $\mathbb{R}_+$ and infinite rates, we have a hierarchy of rates $(i, r) \in \mathbb{N} \times \mathbb{R}_+$, where i is the magnitude of the rate. Intuitively a rate of magnitude $i+1$ is infinitely faster than a rate of magnitude i, and has higher priority. So the race condition is performed only between rates of higher magnitude. This is a hierarchy of rates in the sens that all the rates $(0, r)$ are slower than the rates $(1, r')$, which again are slower than all the rates $(2, r'')$,...

There is also a biochemical motivation for this proposition: biochemical systems often exhibit stochastic behaviors ranging over a very large scale of rates, and so it is common to assume some of them to be instantaneous or on the contrary to neglect some of them. Therefore we introduce the multi-scale π-calculus where rates can be of different order of magnitude, and we extend our expressiveness results to it.

The probabilistic settings. In order to complete the picture we also investigate the relationship between separate and mixed choices in the probabilistic settings. Surprisingly it happens that both kind of choices are equally expressive: we provide an encoding from the separate choice to the mixed one. We also conjecture that the separate choice cannot be encoded either in the output guarder choice nor in the input guarded choice.

The Chapter is organized as follows. In Section 6.2 we discuss the definition of weak stochastic transitions and what properties a stochastic encoding should possess. In Section 6.3 we present the syntax and semantics of the stochastic π-calculus and in Section 6.4 we present our expressiveness results in the stochastic settings. In Section 6.5 we introduce the multi-scale π-calculus and extend our expressiveness results to its settings. In Section 6.6 we introduce the syntax and semantics of the quantitative and probabilistic π-calculi. In Section 6.7 we present our expressiveness results in the probabilistic settings. In Section 6.8 we discuss the relative expressive power of the separate choice and the input and output guarded choices. Finally, in Section 6.9 we discuss the design of our calculus. The Chapter is closed by a conclusion in Section 6.10 and a discussion on related works in Section 6.11.

6.2 On weak stochastic transitions and stochastic encodings.

Weak transitions. In the classical setting, observing all the transitions is often considered as a too precise level of observation. The so-called τ-moves, namely the internal synchronization of a process, are usually considered as non observable. The weak transitions permit us to obtain a more realistic point of view. Let $\longmapsto$ be the one-step transition relation, and $\longmapsto^*$ its reflexive and transitive closure, then the weak-transition relation $\Longrightarrow$ is defined by:

$$P \stackrel{\alpha}{\Longrightarrow} Q \Leftrightarrow P \stackrel{\tau}{\longmapsto}^* \stackrel{\alpha}{\longmapsto} \stackrel{\tau}{\longmapsto}^* Q,$$

where α is a label and $\stackrel{\tau}{\longmapsto}$ is an internal synchronization.

The stochastic setting introduces another important observable: the *elapsing of time*. We consider the finite rate transitions as observable and the instantaneous transitions as unobservable: since these transitions take no time, one cannot tell how many of them have occurred. So we define the weak stochastic transition in the following way:

Definition 6.2.1 (Weak stochastic transitions) *The weak stochastic transition relation is defined by* $P \stackrel{\lambda}{\Longrightarrow} Q \Leftrightarrow P \stackrel{\infty}{\longmapsto}^* \stackrel{\lambda}{\longmapsto} \stackrel{\infty}{\longmapsto}^* Q$.

Alternatively to the previous definition one may eliminate the unobservable instantaneous transitions using the downgrading process, as defined in definition 2.2.3. The reason why we do not follow this direction is exactly because the downgrading process eliminates the non-determinism introduced by the infinite rates, while in this Chapter on the contrary we wish to investigate the consequences in terms of expressiveness of the introduction of rates, both finite or infinite.

Stochastic encodings. What properties do we require for an encoding between stochastic languages ? The answer certainly depends on whether we want to prove a positive result or a separation result. Stronger results are obtained by imposing stronger conditions in the first case and weaker conditions in the latter one. There is not yet a general agreement about which should be considered "good conditions", still one can identify a set of minimum conditions an encoding should satisfy (see [61] for a review on classical encoding criterions). We choose to consider the following conditions.

If P can do an infinite sequence of instantaneous transitions we write $P \stackrel{\infty}{\Longrightarrow}^{\Omega}$.

Definition 6.2.2 (Admissible encoding) *An encoding* $[\![.]\!]$ *is* admissible *if and only if:*

1. *for all processes* P *and* Q, $[\![P|Q]\!] = \delta_{P,Q}([\![P]\!]|[\![Q]\!])$ *where* $\delta_{P,Q}$ *is a possibly empty name declaration;*
2. $P \stackrel{\infty}{\Longrightarrow}^{\Omega}$ *if and only if* $[\![P]\!] \stackrel{\infty}{\Longrightarrow}^{\Omega}$;

3. *for all* $\mu \in \mathbb{R}_+ \bigcup \{\infty\}$, *if* $P \stackrel{\mu}{\Longrightarrow} Q$ *then* $[\![P]\!] \stackrel{\mu}{\Longrightarrow} [\![Q]\!]$;

4. *for all* $\lambda \in \mathbb{R}_+$, *if* $[\![P]\!] \stackrel{\lambda}{\Longrightarrow} R$ *then* $\exists Q$ *such that* $R \stackrel{\infty}{\Longrightarrow} [\![Q]\!] \wedge P \stackrel{\lambda}{\Longrightarrow} Q$.

The reasons for this definitions are the following:

- An encoding should be *uniform* i.e. homomorphic with respect to the parallel composition, namely $[\![P|Q]\!] = \delta_{P,Q}([\![P]\!]|[\![Q]\!])$ where $\delta_{P,Q}$ is a declaration of new names. This ensures that two parallel processes are translated into two parallel processes, possibly using new names but without any coordinator process.

- In the non stochastic setting introducing τ-divergence, that is infinite sequence of τ-moves, is sometimes accepted. However in the stochastic setting such divergence represents a serious problem: since the silent transitions are the instantaneous ones, and since such transitions have priority over the finite rate ones, it means that one will keep doing instantaneous transitions forever without elapsing of time. Therefore we ask that a process can do an infinite sequence of instantaneous transitions if and only if its encoding can do the same.

- An encoding should preserve a "reasonable" semantics. Among the various criterion proposed in [61], we choose the *operational correspondence* since it is one of the weakest and we adapted it to the stochastic setting. More precisely we ask that any weak transition from P to Q is mimicked by the encodings of P and Q, and that any weak transition starting from the encoding of P is the beginning of a weak transition that matches a weak transition of P.

Our separation results are formulated using the notion of admissible encoding. Remarkably this property is preserved by composition:

Proposition 6.2.1 *Given two admissible encodings* $[\![(.)]\!]$ *and* $\{[.]\}$ *their composition* $[\![(\{[.]\})]\!]$ *is also an admissible encoding.*

6.3 The stochastic π-calculi: syntax and semantics.

In this section we present the syntax and semantics of the stochastic π-calculus [64, 65, 67, 66]. Actually the operators are the same as the ones of the `nano`π-calculus (which we defined in the Chapter 4), but here they are allowed to be combined in more flexible ways.

Definition 6.3.1 (Syntax of the stochastic π-calculus) *The stochastic π-calculus uses two set of names: channel names, which are totally ordered, and agent names. Channel names are associated with a rate that is either strictly*

positive or infinite. This rate is either explicitly declared in the process or globally defined (for free names). The terms of the stochastic π-calculus use the following syntactic categories:

$$\begin{array}{lcl} P & ::= & P|P \quad | \quad \sum\limits_{i\in I}\alpha_i.P_i \quad | \quad (\widetilde{x}:\widetilde{\lambda})P \quad | \quad [u=v]P \quad | \quad A(\widetilde{z}) \quad | \quad 0 \\ \alpha & ::= & \overline{x}\,\widetilde{y} \quad | \quad x\,(\widetilde{y}) \end{array}$$

The meaning of these operators has been already discussed in the Section 4.2. We recall that each channel is associated with a rate, which is defined either globally for the free names or with the name restriction operator otherwise, and that for each agent $A(\widetilde{z})$ we ask for an unique definition $A(\widetilde{z}) := P$.

Notation. In the following msπ and ssπ refer to the stochastic π-calculus where all choices are mixed or separate respectively, and where the rates of all channels are finite. The corresponding languages with infinite rates are denoted msπ^∞ and ssπ^∞ respectively.

In order to define the basic transition relation of the stochastic π-calculus we need to define its tagged version. We assume an infinite set of tags that is closed by the concatenation operator "." such that given two tags ξ and χ, $\xi.\chi$ is also a tag. Given a set of tags X we define $\xi.X$ to be the set $\{\xi.\chi \,|\, \chi \in X\}$.

Definition 6.3.2 (The tagged stochastic π-calculus) *Given a term P its tagged version P^* is defined as follows:*

$$\begin{array}{rcl} (P|Q)^* & = & P^*|Q^* \\ (\sum\limits_{i\in I}\alpha_i.P_i)^* & = & \sum\limits_{i\in I}\alpha_i^{\xi_i}.(P_i)^* \\ ((\widetilde{x}:\widetilde{\lambda})P)^* & = & (\widetilde{x}:\widetilde{\lambda})(P)^* \\ ([u=v]P)^* & = & [u=v](P^*) \\ (A(\widetilde{z}))^* & = & A^{\xi}(\widetilde{z}) \end{array}$$

where the ξ_i's are pairwise different fresh tags, where we assume that the set of tags used in P_i^* are pairwise disjoint and similarly for the set of tags of the processes P^* and Q^*.

Given a tagged term Q, the tagged term $\xi.Q$ is the term Q where all tags have been prefixed by ξ.

Given a tagged term P, its untagged version is obtained by erasing all tags and is noted $\overline{P}$.

Definition 6.3.3 *The basic transition relation for the stochastic π-calculus is written $\xrightarrow{\mu}_X$ where X is a set of identifiers and where $\mu \in \{(\widetilde{z}:\widetilde{\eta})\overline{x}\,\widetilde{y}\ ;\ x\,(\widetilde{y})\ ;\ \tau_\lambda\}$, and it is the least relation defined as :*

$$(\texttt{Sum})\ \frac{}{\sum\limits_{\mathtt{i}\in\mathtt{I}} \alpha_{\mathtt{i}}^{\mathtt{x_i}}.\mathtt{P_i} \xrightarrow{\alpha_{\mathtt{i}}}_{\mathtt{x_i}} \mathtt{P_i}} \qquad (\texttt{Par})\ \frac{\mathtt{P} \xrightarrow{\mu}_{\mathtt{X}} \mathtt{Q} \quad \mathtt{bn}(\mu) \bigcap \mathrm{name}(\mathtt{R}) = \emptyset}{\mathtt{P}|\mathtt{R} \xrightarrow{\mu}_{\mathtt{X}} \mathtt{Q}|\mathtt{R}}$$

$$(\texttt{New})\ \frac{\mathtt{P} \xrightarrow{\mu}_{\mathtt{X}} \mathtt{Q} \quad \mathtt{z} \notin \mathrm{name}(\mu)}{(\mathtt{z}:\lambda)\mathtt{P} \xrightarrow{\mu}_{\mathtt{X}} (\mathtt{z}:\lambda)\mathtt{Q}} \qquad (\texttt{Open})\ \frac{\mathtt{P} \xrightarrow{(\widetilde{\mathtt{y}}:\widetilde{\eta})\overline{\mathtt{x}}\,\widetilde{\mathtt{u}}}_{\mathtt{X}} \mathtt{Q} \quad \mathtt{z} \neq \mathtt{x} \quad \mathtt{z} \notin \widetilde{\mathtt{u}} \setminus \widetilde{\mathtt{y}} = \emptyset}{(\mathtt{z}:\lambda)\mathtt{P} \xrightarrow{(\mathtt{z}:\lambda,\widetilde{\mathtt{y}}:\widetilde{\eta})\overline{\mathtt{x}}\,\widetilde{\mathtt{u}}}_{\mathtt{X}} \mathtt{Q}}$$

$$(\texttt{Com})\ \frac{\mathtt{P} \xrightarrow{(\widetilde{\mathtt{y}}:\widetilde{\eta})\overline{\mathtt{x}}\,\widetilde{\mathtt{u}}}_{\mathtt{X}} \mathtt{P}' \quad \mathtt{Q} \xrightarrow{\mathtt{x}\,(\widetilde{\mathtt{v}})}_{\mathtt{Y}} \mathtt{Q}' \quad \mathtt{fn}(\mathtt{Q}) \bigcap \widetilde{\mathtt{y}}}{(\widetilde{\mathtt{y}}:\widetilde{\eta})(\mathtt{P}|\mathtt{Q}) \xrightarrow{\tau_{rate(\mathtt{x})}}_{\mathtt{X}\bigcup\mathtt{Y}} (\widetilde{\mathtt{y}}:\widetilde{\eta})(\mathtt{P}|\mathtt{Q}\{\widetilde{\mathtt{u}}/\widetilde{\mathtt{v}}\}}$$

$$(\texttt{Ag})\ \frac{\mathtt{P}^*\{\widetilde{\mathtt{u}}/\widetilde{\mathtt{v}}\} \xrightarrow{\mu}_{\mathtt{X}} \mathtt{Q} \quad \mathtt{A}(\widetilde{\mathtt{v}}) := \mathtt{P}}{\mathtt{A}^{\alpha}(\widetilde{\mathtt{u}}) \xrightarrow{\mu}_{\alpha.\mathtt{X}} \alpha.\mathtt{Q}}$$

plus the symmetric version of the (Par) and (Com) rules.

Definition 6.3.4 *The structural congruence of the stochastic π-calculus is the least equivalence relation such that:*

- $P \equiv Q$ *if* Q *is obtained from* P *by* α*-renaming,*
- $P|Q \equiv Q|P$ *and* $(P|Q)|R \equiv P|(Q|R)$,
- $\sum\limits_{i\in I} \alpha_i^{\xi_i}.P_i \equiv \sum\limits_{\sigma(i)\in I} \alpha_i^{\xi_i}.P_i$ *for any permutation* σ *of* I,
- $(x:\lambda)(P|Q) \equiv P|(x:\lambda)Q$ *when* $x \notin \mathtt{fn}(P)$,
- $(x:\lambda)0 \equiv 0$,
- $0 + P \equiv 0|P \equiv P$,
- $(x:\lambda)(y:\eta)P \equiv (y:\eta)(x:\lambda)P$.

Then the collective transition of the stochastic π-calculus is built from the basic transition relation and the structural congruence according to the definition 2.1.2.

The version of the stochastic π-calculus presented here differs from the original one [64, 65, 67, 66] where rates are attached to actions rather than to channel names. We choose to study this version rather than the other one because as far as we know it is nowadays much more used, especially in the formal system biology field.

6.4 Gaps and bridges between synchronous and asynchronous stochastic π-calculi.

In this section we collect our expressiveness results. We prove that without infinite rate mixed choice is strictly more expressive than separate choice (Part

6.4.1), we extend this result to the case of infinite rates (Part 6.4.2) and we show that separate choice with infinite rates can encode finite rate mixed choice assuming that (Part 6.4.3).

6.4.1 The msπ calculus is strictly more expressive than the ssπ calculus.

Before stating our first result we need to prove the following key lemma:

Lemma 6.4.1 *$\forall P \in ss\pi$, if P can be written $P = (\widetilde{x} : \widetilde{r})(Q|Q)$ and if $P \stackrel{\lambda}{\longmapsto} P'$ then $\exists P'' \in ss\pi$ such that $P' \stackrel{\lambda'}{\longmapsto} P''$.*

Proof Such a collective transition $P \stackrel{\lambda}{\longmapsto} P'$ corresponds to a set of basic transitions which are either internal to Q or are synchronizations between the two Q's.

If there is one transition of the first type, then it means that $P' \equiv (\widetilde{y} : \widetilde{t})(Q'|Q)$, with $Q \stackrel{\lambda}{\longmapsto} Q'$. Since P' contains another Q it can do another transition.

Otherwise there is one transition of the second type, and it means that Q contains an input and an output on a same channel, say x. Since Q has only separate choice, this implies that $Q \equiv (\widetilde{y} : \widetilde{s})(R \,|\, \overline{x}\, z.Q_1 + \Sigma \,|\, x\,(w).Q_2 + \Sigma')$, where Σ and Σ' are the remaining part of the choices. Thus $P' \equiv (\widetilde{z} : \widetilde{t})(R \,|\, Q_1 \,|\, x\,(w).Q_2 + \Sigma' \,|\, R \,|\, \overline{x}\, z.Q_1 + \Sigma \,|\, Q_2)$, from which a synchronization on the channel x is again possible. □

We can now state and prove the first result of this section:

Theorem 6.4.1 *There is no admissible encoding from msπ to ssπ.*

Proof By restricting the encoding of the Chapter 4 to finite rates, we get an encoding from `nano`κ to msπ. The correctness Theorem 4.5.4 states that this is an admissible encoding. Suppose there exists an admissible encoding from msπ to ssπ. Then we can compose it with the one from `nano`κ to msπ, to obtain an encoding from `nano`κ to ssπ. By Proposition 6.2.1 it is again an admissible encoding. We now prove that no such encoding exists, which ends the proof.

Suppose then that an admissible encoding $[\![.]\!]$ from `nano`κ to ssπ exists. Since in this case there is no instantaneous transitions, the items 3 and 4 of the admissibility definition imply that $P \stackrel{\lambda}{\longmapsto} Q \Leftrightarrow [\![P]\!] \stackrel{\lambda}{\longmapsto} [\![Q]\!]$. Consider the system where there are two species A and A' with no field and no site, and only one reaction: $A, A \stackrel{\lambda}{\longmapsto} A', A'$. And consider the encoding of the solution $S = A, A$ and $S' = A', A'$. Since $S \stackrel{\lambda}{\longmapsto} S'$, $[\![S]\!] \stackrel{\lambda}{\longmapsto} [\![S']\!]$. Since the encoding is uniform, $[\![S]\!]$ can be written $(\widetilde{x} : \widetilde{t})(Q|Q)$ where $Q = [\![A]\!]$. So by Lemma 6.4.1 this implies that there exists a transition $[\![S']\!] \stackrel{\lambda'}{\longmapsto} T$. But the solution S' cannot match this transitions since no reaction can be applied in it. □

6.4.2 The msπ^∞ calculus is strictly more expressive than the ssπ^∞ calculus.

We carry on with the extension of the previous result to the π-calculi with infinite rates. In this case we need to consider a coarser problem to obtain our separation result. Before stating it, we need to introduce the concept of *networks* (originally used in [60]).

Formally a network of processes of the π-calculus is a term $(\widetilde{x} : \widetilde{\lambda})\langle\!\langle P_1; \ldots; P_n \rangle\!\rangle$ where the scope of the restriction on $\widetilde{x}$ is the set of the P_i's. It represents the process $(\widetilde{x} : \widetilde{\lambda})(P_1 | \ldots | P_n)$ while explicitly keeping track of the distribution of the processes. Indeed the term $P|Q|R$ may be interpreted as the parallel composition of three processes P, Q and R or as the composition of two processes $P|Q$ and R. In networks this ambiguity is removed: in $\delta\langle\!\langle P_1; \ldots; P_n \rangle\!\rangle$ each P_i represents a separate process.

Next we add the commutativity of ";" to the structural congruence: that is $\langle\!\langle P; Q \rangle\!\rangle \equiv \langle\!\langle Q; P \rangle\!\rangle$.

The counterpart of the rule (Open) and (New) for the networks are added to basic transition as well as the following three rules and the symmetric version of the last two:

$$(\mathtt{Intro-Net})\ \frac{\mathtt{P} \xrightarrow{\mu}_{\mathtt{X}} \mathtt{Q}}{\langle\!\langle \mathtt{P} \rangle\!\rangle \xrightarrow{\mu}_{\mathtt{X}} \langle\!\langle \mathtt{Q} \rangle\!\rangle}$$

$$(\mathtt{Par-Net})\ \frac{\langle\!\langle \mathtt{P} \rangle\!\rangle \xrightarrow{\mu}_{\mathtt{X}} \langle\!\langle \mathtt{Q} \rangle\!\rangle \quad \mathtt{bn}(\mu) \bigcap \mathrm{name}(\langle\!\langle \mathtt{R_1}; \ldots; \mathtt{R_n} \rangle\!\rangle) = \emptyset}{\langle\!\langle \mathtt{P}; \mathtt{R_1}; \ldots; \mathtt{R_n} \rangle\!\rangle \xrightarrow{\mu}_{\mathtt{X}} \langle\!\langle \mathtt{Q}; \mathtt{R_1}; \ldots; \mathtt{R_n} \rangle\!\rangle}$$

$$(\mathtt{Com-Net})\ \frac{\langle\!\langle \mathtt{P} \rangle\!\rangle \xrightarrow{(\widetilde{\mathtt{y}}:\widetilde{\eta})\overline{\mathtt{x}}\,\widetilde{\mathtt{u}}}_{\mathtt{X}} \langle\!\langle \mathtt{P'} \rangle\!\rangle \quad \langle\!\langle \mathtt{Q} \rangle\!\rangle \xrightarrow{\mathtt{x}\,(\widetilde{\mathtt{v}})}_{\mathtt{Y}} \langle\!\langle \mathtt{Q'} \rangle\!\rangle \quad \mathtt{fn(Q)} \bigcap \widetilde{\mathtt{y}}}{(\widetilde{\mathtt{y}} : \widetilde{\eta})\langle\!\langle \mathtt{P}; \mathtt{Q} \rangle\!\rangle \xrightarrow{\tau_{rate(\mathtt{x})}}_{\mathtt{X} \bigcup \mathtt{Y}} (\widetilde{\mathtt{y}} : \widetilde{\eta})\langle\!\langle \mathtt{P}; \mathtt{Q}\{\widetilde{\mathtt{u}}/\widetilde{\mathtt{v}}\} \rangle\!\rangle}$$

Then the collective transition relation is defined as usually (see Definition 2.1.2).

We say that an admissible encoding is *distributed* when the encoding of a solution $\mathsf{S} = A_1[u_1](\sigma_1), \ldots, A_n[u_n](\sigma_n)$ is a network $\delta_{\mathsf{S}}\langle\!\langle [\![A_1[u_1](\sigma_1)]\!]; \ldots; [\![A_n[u_n](\sigma_1 n)]\!] \rangle\!\rangle$. The criterion used to prove our separation result is now:

> *Does there exist an admissible and distributed encoding from the* nanoκ *calculus with infinite rates and where reaction neither create nor destroy molecules to a fragment of the stochastic π-calculus?*

Similarly to the previous section we start with a key lemma. It differs from the lemma 6.4.1 by the fact that it concerns instantaneous transitions and networks and because it requires that P'' is also a symmetric process.

Lemma 6.4.2 *$\forall Q \in ss\pi^\infty$, let P be the network $(\widetilde{x} : \widetilde{r})\langle\!\langle Q; Q \rangle\!\rangle$. If $P \overset{\infty}{\longmapsto} P'$ then $\exists Q' \in ss\pi^\infty$ and $(\widetilde{y} : \widetilde{s})$ such that $P' \overset{\infty}{\longmapsto} (\widetilde{y} : \widetilde{s})\langle\!\langle Q'; Q' \rangle\!\rangle$.*

Proof Such a transition $P \stackrel{\infty}{\longmapsto} P'$ corresponds either to an instantaneous transition internal to Q or to an instantaneous synchronization between the two Qs.

In the first case, P' can be written $(\widetilde{z} : \widetilde{t})(\!|Q'; Q|\!)$, with $Q \stackrel{\infty}{\longmapsto} Q'$. So $P' \stackrel{\infty}{\longmapsto} (\widetilde{z'} : \widetilde{t'})(\!|Q'; Q'|\!)$.

In the second case, Q contains an input and an output on the same channel, say x, whose rate is infinite. Since Q has only separate choice, this implies that Q can be written $(\widetilde{y} : \widetilde{t})(R \,|\, \overline{x}\, z.Q_1 + \Sigma \,|\, x\,(w).Q_2 + \Sigma')$ where Σ and Σ' are the remaining part of the choices. Thus P' can be written $(\widetilde{z} : \widetilde{s})(\!|R \,|\, Q_1 \,|\, x\,(w).Q_2 + \Sigma' \,;\, R \,|\, \overline{x}\, z.Q_1 + \Sigma \,|\, Q_2\{z/w\}|\!)$. And so $P' \stackrel{\infty}{\longmapsto} (\widetilde{z'} : \widetilde{t'})(\!|R \,|\, Q_1 \,|\, Q_2\{z/w\} \,;\, R \,|\, Q_1 \,|\, Q_2\{z/w\}|\!)$. It suffices now to define $Q' = R \,|\, Q_1 \,|\, Q_2\{z/w\}$. □

Theorem 6.4.2 *There is no admissible encoding from msπ^∞ to ssπ^∞.*

Proof First we remark that we can restrict the encoding of the Chapter 4 to reactions which neither create nor destroy molecules. In this way we get an encoding from nanoκ (with infinite rates and where reactions neither create nor destroy molecules) to msπ^∞. The correctness Theorem 4.5.4 guarantees that this is admissible. Actually it is also distributed. Indeed if we write the encoding of a solution $\mathsf{S} = A_1[u_1](\sigma_1), \ldots, A_n[u_n](\sigma_n)$ as the following network $\delta_S(\!|\langle A_1[u_1](\sigma_1)\rangle, \ldots, (\!|A_n[u_n](\sigma_n)|\!)|\!)$, we can prove again the same correctness theorem. Let us now suppose that there is an admissible encoding from msπ^∞ to ssπ^∞. We can compose it with the one from nanoκ to msπ^∞ and get an encoding from nanoκ to ssπ^∞: in the above network each process msπ^∞ $\langle A_1[u_1](\sigma_1)\rangle$ is replaced by its encoding, which belongs to ssπ^∞. By Proposition 6.2.1 this yields an admissible encoding and by construction it is also distributed. We now prove that no such encoding exists, which yields the theorem.

Suppose then that a distributed and admissible encoding $[\![.]\!]$ from nanoκ to ssπ^∞ exists. Consider the system where there are three species A, A' and A'' with no field and no site, and two reactions: $A, A \stackrel{\infty}{\longmapsto} A', A''$ and $A, A' \stackrel{\lambda}{\longmapsto} A'', A'$. And consider the encoding of the solution $S = A, A$ and $S' = A', A''$. Since the encoding is admissible and distributed, $[\![S]\!] = (\widetilde{x} : \widetilde{t})\langle Q; Q\rangle$ where $Q = [\![A]\!]$. Since $S \stackrel{\infty}{\longmapsto} S'$, there exists R such that $[\![S]\!] \stackrel{\infty}{\Longrightarrow} R$. By Lemma 6.4.2 this implies that there exists a Q' such that $R \stackrel{\infty}{\longmapsto} (\widetilde{x'} : \widetilde{t'})\langle Q'; Q'\rangle$. If this new process can again do an instantaneous transition the lemma can be applied again, possibly infinitely many times depending on the processes. In this way we build two sequences of processes $(R_i)_{i\in I}$ and $(Q_i)_{i\in I}$ such that $Q_0 = Q$, $R_0 = R$ and $\forall i \in I$ there exists a declaration of name δ_i such that $\delta_i\langle Q_i; Q_i\rangle \stackrel{\infty}{\longmapsto} R_i \stackrel{\infty}{\longmapsto} \delta_{i+1}\langle Q_{i+1}; Q_{i+1}\rangle$. Because the solution S does not diverge and since admissible encodings preserve divergence, the set I is finite. Let i_0 be its maximum. We have that $[\![S]\!] \stackrel{\infty}{\Longrightarrow} \delta_{i_0}\langle Q_{i_0}; Q_{i_0}\rangle$, so by the last item of the admissibility definition there exists a solution T such that $\delta_{i_0}\langle Q_{i_0}; Q_{i_0}\rangle \stackrel{\infty}{\Longrightarrow} [\![T]\!]$ and $S \stackrel{\infty}{\Longrightarrow} T$. By maximality of i_0, and because there is only one instantaneous transition from S, we obtain that $\delta_{i_0}\langle Q_{i_0}; Q_{i_0}\rangle = [\![A', A'']\!]$. And so $[\![A']\!] = [\![A'']\!] = Q_{i_0}$.

We now have a contradiction because A' can perform a reaction which A'' cannot mimic, and thus their encodings must be different. □

6.4.3 The ssπ^∞ calculus versus the msπ calculus.

We now consider the question whether the addition of infinite rates to the separate choice allows encoding the mixed choice. Unfortunately we are not able to provide a full answer. However if we assume mixed choices to be proper (that is without input and output on the same channel) then the encoding is possible. This is presented in the following definition:

Definition 6.4.1 *Let $[\![.]\!]$ be the encoding from the proper mixed choice π-calculus with finite rates to ssπ^∞ defined on the proper mixed choice by:*

$$\begin{aligned}[\![\sum_{i\in I}\overline{x_i}\,y_i.P_i + \sum_{j\in J}x_j\,(y_j).Q_j]\!] \quad &\triangleq \quad (z:\infty, z':\infty)\\ &\quad [\,\sum_{i\in I}\overline{x_i}\,y_i.([\![P_i]\!]\,|\,\overline{z'}\,) + \overline{z}\\ &\quad |\,\sum_{j\in J}x_j\,(y_j).([\![Q_j]\!]\,|\,z) + z'\,]\end{aligned}$$

and defined homomorphically on the other operators, that is $[\![P|Q]\!] = [\![P]\!]|[\![Q]\!]$, $[\![(\widetilde{x}:\widetilde{\eta})P]\!] = (\widetilde{x}:\widetilde{\eta})[\![P]\!]$ and $[\![[u=v]P]\!] = [u=v][\![P]\!]$, and each equation $A(\widetilde{x}) := P$ becomes $[\![A(\widetilde{x})]\!] := [\![P]\!]$.

Intuitively the mixed choice is divided into its input-guarder and output-guarded components that are put in parallel. In order to ensure that after a synchronization, on the output $\overline{x_i}\,y_i$ for instance, the input choice is disabled we develop a preemption mechanism. Once the output has been consumed the continuation $[\![P_i]\!]$ is put at top level as expected, but together with an output on the channel z'. This output can synchronize with the corresponding input in the input choice, and since it has infinite rate, this synchronization has priority on any other transition.

This encoding would not work if the mixed choice was not proper, that it if it has an input and an output on the same channel, because these actions would be able to synchronize in the encoding. The following theorem states the correctness of this encoding.

Theorem 6.4.3 *For all terms P of the proper mixed choice π-calculus with finite rates, and for all $\lambda \in \mathbb{R}\bigcup\{\infty\}$:*

1. *If $P \stackrel{\lambda}{\longmapsto} Q$ then $[\![P]\!] \stackrel{\lambda}{\longmapsto}\!\!\stackrel{\infty}{\Longrightarrow} [\![Q]\!]$.*
2. *If $[\![P]\!] \stackrel{\lambda}{\longmapsto} R$ then there exists a term Q of the proper mixed choice π-calculus such that $R \stackrel{\infty}{\Longrightarrow} [\![Q]\!]$ and $P \stackrel{\lambda}{\longmapsto} Q$.*

Theorem 6.4.3 concerns the collective transition relation. In order to prove it we first need to prove a lemma reformulating the theorem for the basic transition relation:

Lemma 6.4.3 *For all term P of the proper mixed choice π-calculus:*

1. *If* $P \xrightarrow{\mu}_X Q$ *then there exists* R *and* R' *such that* $[\![P]\!] \xrightarrow{\mu}_X R \stackrel{\infty}{\Longrightarrow} R' \equiv [\![Q]\!]$.

2. *If* $[\![P]\!] \xrightarrow{\mu}_X R$ *then there exists* Q *and* R' *such that* $R \stackrel{\infty}{\Longrightarrow} R' \equiv [\![Q]\!]$ *and* $P \xrightarrow{\mu}_X Q$.

Proof

1. The result is obtained by an induction on the proof tree of $P \xrightarrow{\mu}_X Q$. If the transition has been obtained by the (Sum) rule, let us suppose without loss of generality that μ is an input, it means that P is a choice written $\sum_{i\in I}(x_i\,(\widetilde{z_i}))^{a_i}.P_i + \sum_{j\in J}(\overline{y_j}\,\widetilde{z_j})^{b_j}.P_j$ and there is a i_0 be such that $\mu = x_{i_0}\,(\widetilde{z_{i_0}})$, $X = \{a_{i_0}\}$ and $Q = P_{i_0}$. Then by definition of the encoding:

$$\begin{array}{rcl}[\![P]\!] & = & (z:\infty, z':\infty)(\\ & & \sum_{i\in I}(x_i\,(\widetilde{z_i}))^{a_i}.([\![P_i]\!]|(z)^{d_i}) + (z')^e \\ & & |\ \sum_{j\in J}(\overline{y_j}\,\widetilde{z_j})^{b_j}.([\![P_j]\!]|(\overline{z'})^{c_j}) + (\overline{z})^f)\end{array}$$

And so:

$$\begin{array}{rcll}[\![P]\!] & \stackrel{x_{i_0}\,(\widetilde{z_{i_0}})}{\longrightarrow}_{a_{i_0}} & (z:\infty, z':\infty)(& \\ & & [\![P_{i_0}]\!]|(z)^{d_i} & \\ & & |\ \sum_{j\in J}(\overline{y_j}\,\widetilde{z_j})^{b_j}.([\![P_j]\!]|(\overline{z'})^{c_j}) + (\overline{z})^f) & \\ & \stackrel{\infty}{\longmapsto} & [\![P_{i_0}]\!] & \end{array}$$

where the second transition is obtained by synchronization on the channel z. Thus we have $[\![P]\!] \xrightarrow{\mu}_X \stackrel{\infty}{\longmapsto} [\![Q]\!]$ since $P_{i_0} = Q$.

The other cases are easily proved using the fact that the encoding is homomorphic on every operators but the choice.

2. The result is obtained by induction on the structure of P. If $[\![P]\!]$ is a choice $\sum_{i\in I}(x_i\,(\widetilde{z_i}))^{a_i}.P_i + \sum_{j\in J}(\overline{y_j}\,\widetilde{z_j})^{b_j}.P_j$, then by definition of the encoding:

$$\begin{array}{rcl}[\![P]\!] & = & (z:\infty, z':\infty)(\\ & & \sum_{i\in I}(x_i\,(\widetilde{z_i}))^{a_i}.([\![P_i]\!]|(z)^{d_i}) + (z')^e \\ & & |\ \sum_{j\in J}(\overline{y_j}\,\widetilde{z_j})^{b_j}.([\![P_j]\!]|(\overline{z'})^{c_j}) + (\overline{z})^f)\end{array}$$

Since we assume that the mixed choice are proper no synchronization can

occur between the x_i and the y_j, so any transition of $[\![P]\!]$ is of the form:

$$[\![P]\!] \quad \stackrel{x_{i_0}\,(\widetilde{z_{i_0}})}{\longrightarrow}_{a_{i_0}} \quad (z:\infty, z':\infty)(\\ [\![P_{i_0}]\!]|(z)^{d_i} \\ |\sum_{j\in J}(\overline{y_j}\,\widetilde{z_j})^{b_j}.([\![P_j]\!]|(\overline{z'})^{c_j}) + (\overline{z})^f)$$

or similarly in case of an output. So scheduling the synchronization on the channel z we obtain:

$$(z:\infty, z':\infty)([\![P_{i_0}]\!]|(z)^{d_i}|\sum_{j\in J}(\overline{y_j}\,\widetilde{z_j})^{b_j}.([\![P_j]\!]|(\overline{z'})^{c_j}) + (\overline{z})^f) \stackrel{\infty}{\longmapsto} \equiv [\![P_{i_0}]\!]$$

And finally it suffices to remark that $P \stackrel{x_{i_0}\,(\widetilde{z_{i_0}})}{\longrightarrow}_{a_{i_0}} P_{i_0}$.

Again, the other cases are easily proved using the fact that the encoding is homomorphic on every operators but the choice.

□

We can now prove Theorem 6.4.3:
Proof

1. Suppose that $P \stackrel{\lambda}{\longmapsto} Q$. By construction of the collective transition relation there is a basic transition $P^* \stackrel{\tau_\eta}{\longrightarrow}_X Q^*$ and $\lambda = \sum_{((\tau_\eta,X),R)\in[next(P^*)]_{Q^*}} \eta$.
   As a consequence of Lemma 6.4.3:

$$\sum_{((\tau_\eta,X),R)\in[next(P^*)]_{Q^*}} \eta = \sum_{((\tau_\eta,X),[\![R]\!])\in[next([\![P^*]\!])]_{[\![Q^*]\!]}} \eta$$

   and so by construction of the collective transition relation, $[\![P]\!] \stackrel{\lambda}{\longmapsto}\stackrel{\infty}{\Longrightarrow} [\![Q]\!]$.

2. Suppose that $[\![P]\!] \stackrel{\lambda}{\longmapsto} R$. By construction of the collective transition relation, there is a basic transition $[\![P]\!]^* \stackrel{\lambda}{\longmapsto} R^*$ and we have that $\lambda = \sum_{((\tau_\eta,X),R'^*)\in[next([\![P]\!]^*)]_{R^*}} \eta$. By lemma 6.4.3:

$$\sum_{((\tau_\eta,X),\overline{R'^*})\in[next(P)^*]_{R^*}} \eta = \sum_{((\tau_\eta,X),R'^*)\in[next([\![P^*]\!])]_{R^*}} \eta$$

   and so by construction of the collective transition relation, $P \stackrel{\lambda}{\longmapsto}\stackrel{\infty}{\Longrightarrow} Q$.

□

Interestingly, we have the following corollary:

Corollary 6.4.1 *The encoding of Definition 6.4.1 is admissible.*

6.5 Generalization of the results to the case of the multi-scale π-calculus.

In this section we introduce the *multi-scale* π-calculus: it extends the stochastic π-calculus with rates of different order of magnitude. Two main reasons motivate this proposition. First, biochemical systems often exhibit stochastic behaviors ranging over a very large scale of rates. It is common to assume some of them to be instantaneous or on the contrary to neglect some of them. The multi-scale π-calculus is a first attempt to cope with such phenomena. Moreover the results of the previous section and in particular the result of part 6.4.3 suggest to introduce several degrees of infinity in the rates.

Definition 6.5.1 (Syntax of the multi-scale π-calculus) *The multi-scale π-calculus uses two set of names: channel names, which are totally ordered, and agent names. Channel names are associated with a rate that is a pair $(i, r) \in \mathbb{Z} \times \mathbb{R}_+$. This rate is either explicitly declared in the process or globally defined (for free names). The terms of the multi-scale π-calculus use the same syntactic categories as the stochastic π-calculus (see Definition 6.3.1).*

Our proposition is to replace rates, which are strictly positive or infinite, by a pair (i, r) where i is a relative integer and r is a strictly positive real number. The order of magnitude of the rate is modeled by i, the higher i the higher the order of magnitude. A transition whose rate is (i, r) has priority on every transition of rate (j, s) such that $j < i$. Such priority are not unheard of (see EMPA language for instance [9]) but as far as we know it is the first time it is used in terms of order of magnitude.

This new definition of rates does not affect the tagged version, the structural congruence and the basic transition relation of the multi-scale π-calculus:

Definition 6.5.2 *The tagged version of the multi-scale π-calculus, its structural congruence and its basic transition relation are defined exactly as the ones of the stochastic π-calculus (see Definitions 6.3.2, 6.1 and 6.3.3).*

On the contrary the definition of the collective transition relation needs to be adapted. In particular the race condition should occur only between the transitions whose rate are of maximal order of magnitude. Before presenting the new collective transition relation we need a few definitions:

- The sum $(i, r) + (i, s)$ is defined to be $(i, r + s)$;
- We define $(i, r) \geq_{ms} (j, s)$ if and only if $i \geq j$;
- $next_{ms}(F) = \{((\mu, \partial), G) \mid F^* \xrightarrow{\mu}_{\partial} G \wedge \forall F^* \xrightarrow{\mu'}_{\partial'} G', rate(\mu) \geq_{ms} rate(\mu')\}$.

The following definition defines the new stochastic transition relation and is parametric in the basic transition relation and the structural congruence:

Definition 6.5.3 *The stochastic transition relation* $\stackrel{\lambda}{\longmapsto}$ *induced by a basic transition relation* $\stackrel{\rho}{\longrightarrow}_\partial$ *and a structural equivalence* $\equiv$ *on a language is the least relation satisfying the following property. Suppose that* $F^* \stackrel{\rho}{\longrightarrow}_\partial G$ *then:*

$$F \stackrel{\lambda}{\longmapsto} can([next_{ms}(F)]_G), \text{ where}$$

$$\lambda = \sum_{((\rho,\partial),G') \in [next(F)]_G,\, rate(\rho)=(i,r)} r$$

We note msπ^n and ssπ^n the mixed choice and separate choice fragments of the multi-scale π-calculus in which the degree of the rates is bounded by n, and we note msπ^Ω and ssπ^Ω the mixed choice and separate choice fragments of the multi-scale π-calculus in which the degree of the rates is not bounded. The extension of the results of the previous section to the multi-scale π-calculus are presented in the following theorem:

Theorem 6.5.1 *For any relative integer n:*

1. *there is no admissible encoding from msπ^n to ssπ^n;*
2. *there is an encoding $[\![.]\!]$ from proper msπ^n to ssπ^{n+1} such that:*
   - *if $P \stackrel{\lambda}{\longmapsto} Q$ then $[\![P]\!] \stackrel{\lambda}{\longmapsto}\stackrel{\infty}{\Longrightarrow} [\![Q]\!]$;*
   - *and if $[\![P]\!] \stackrel{\lambda}{\Longrightarrow} R$ then there exists Q such that $R \stackrel{\infty}{\Longrightarrow} [\![Q]\!]$ and $P \stackrel{\lambda}{\longmapsto} Q$.*
3. *there is an encoding $[\![.]\!]$ from proper msπ^Ω to ssπ^Ω such that:*
   - *if $P \stackrel{\lambda}{\longmapsto} Q$ then $[\![P]\!] \stackrel{\lambda}{\longmapsto}\stackrel{\infty}{\Longrightarrow} [\![Q]\!]$;*
   - *and if $[\![P]\!] \stackrel{\lambda}{\Longrightarrow} R$ then there exists Q such that $R \stackrel{\infty}{\Longrightarrow} [\![Q]\!]$ and $P \stackrel{\lambda}{\longmapsto} Q$.*

Proof

1. The result is proved similarly to Theorem 6.4.1.
2. The encoding is obtained similarly to the one of the part 6.4.3.
3. The encoding is obtained similarly to the one of the part 6.4.3.

□

Conjecture I claim that the following holds:

"There is no admissible encoding from msπ^Ω to proper msπ^Ω."

And consequently, there is no admissible encoding from msπ^Ω to ssπ^Ω.

6.6 An operational semantic of the quantitative and probabilistic π-calculus with mixed choice.

In this section we present two operational semantics for the probabilistic π-calculus with mixed choice: the *quantitative semantics* and the *probabilistic semantics*. They exhibit both probabilistic and non-deterministic behaviors. The variations of behavior due to the environment, the scheduler or the adversary, may be more naturally considered as non-deterministic, since we may not have any information on it a priori. This is why, as in [41], we consider a syntax with a probabilistic choice and a classical parallel operator. The former represents the behavioral variations due to the process in a given scheduling scenario. The latter generates non-deterministically different scheduling scenarios. Correspondingly, the operational semantics yields for each non-deterministic scenario a group of transitions describing the relative probabilities of the possible behaviors in this scenario.

6.6.1 Syntax and semantics of the quantitative and probabilistic π-calculi.

Syntax.

Definition 6.6.1 *The syntax of the probabilistic and quantitative π-calculus is defined by the following grammar*

$$\textit{Prefixes} \quad \alpha ::= \quad \overline{x}y \quad | \quad x(y) \quad | \quad \tau$$

$$\begin{aligned} \textit{Processes} \quad P ::= \quad & \Sigma_{i\in I}(\alpha_i, p_i).P_i \quad | \quad (x)P \quad | \quad P|P \\ & | \quad X \quad | \quad rec_X.P \quad | \quad 0 \end{aligned}$$

where the p_i's are positive real numbers.

The main modification of the syntax is the addition of a positive coefficient on each branch of a choice: the *weights*. This is the method used in [41, 64] for instance. Intuitively, the greater the coefficient of a branch is, the more probable is its execution. We explain in Section 6.6.1 how probabilities are computed from these coefficients. We also adopt a variant of recursion: instead of agent invocation we use recursion variable and recursive process: $rec_X.P$ is the recursive process that defines the recursion variable by $X := P$ where P may contain occurrences of X. We choose this variant because our encoding works much better in this settings.

We sometimes write $(\alpha_1, p_1).P_1 + ... + (\alpha_n, p_n).P_n$ instead of $\Sigma_{i\leq n}(\alpha_i, p_i).P_i$. The empty sum represents a terminated process and is denoted by 0.

In the case of the probabilistic semantics we require that, for each choice $\Sigma_{i\leq n}(\alpha_i, p_i).P_i$, the sum $\Sigma_{i\in I}p_i$ be 1.

Following the terminology used in Concurrency Theory, we call *asynchronous* the sub-language in which outputs can only be followed by the terminated process (*asynchronous outputs*), and occur only in trivial choices, i.e. in choices of cardinality 1. In other words, outputs can only appear in constructs of the form $\overline{x}y.0$.

We use the same structural congruence as for the stochastic π-calculus (see Definition 6.1).

Definition 6.6.2 *The structural congruence of the probabilistic and quantitative π-calculus is the least equivalence relation such that:*

- $P \equiv Q$ *if* Q *is obtained from* P *by* α*-renaming*,
- $P|Q \equiv Q|P$ *and* $(P|Q)|R \equiv P|(Q|R)$,
- $\sum_{i\in I}(\alpha_i, p_i).P_i \equiv \sum_{\sigma(i)\in I}(\alpha_i, p_i).P_i$ *for any permutation* σ *of* I,
- $(x)(P|Q) \equiv P|(x)Q$ *when* $x \notin \mathtt{fn}(P)$,
- $(x)0 \equiv 0$,
- $0 + P \equiv 0|P \equiv P$,
- $(x)(y)P \equiv (y)(x)P$.

Operational Semantics. Weighted choices associate a coefficient with each action. Intuitively this coefficient represents the chances for the action to be executed: the higher the coefficient is, the more probable the action is.

To obtain probabilities from these coefficients, they only need to sum up to 1. It is therefore sufficient to re-normalize each coefficient. Here we have two alternatives: re-normalizing each choice, directly in the term and at each step of execution, or re-normalizing after the derivation of the transitive closure of each possible step. Both possibilities seem reasonable and they correspond to our two semantics: the *probabilistic* semantics and the *quantitative* one, respectively. The probabilistic case requires the sum of the probabilities associated with each step to be equal to 1, and that each rule keeps the sum equal to 1.

The difference between these two semantics is that the coefficient in the quantitative case may be used to represent also other kinds of information. For instance, it could be associated to the expected speed of some reaction which enables the guard (i.e. the inverse of the expected time that takes for the guard to become enabled). As an example, let $P = (a_1, 10).P_1 + (a_2, 10).P_2$ and $Q = (b_1, 5).Q_1 + (b_2, 5).Q_2$, and consider their parallel composition $P \,|\, Q$. In the probabilistic case, all coefficients are re-normalized to $1/2$ and have equal chance to occur. But in the quantitative case, a_1 and a_2 are more likely to occur first. Furthermore b_1 and b_2 could be in competition with a_1 and a_2, for instance if they all are input actions on a channel where there is only an output available. So, the probability to occur first may translate into the probability to occur at all.

Another motivation for considering the quantitative approach is the following: our main result, the encoding of probabilistic mixed choice into probabilistic separate choice, presents some technical problems with respect to the restriction operator in the probabilistic semantics, while this problem disappears in the quantitative semantics.

Rules common to both semantics. As mentioned above the operational semantics do not derive isolated transitions such as $P \stackrel{\alpha}{\longmapsto} Q$ but rather they derive groups of transitions such as $P\{\stackrel{\alpha_i}{\longmapsto}_{p_i} P_i\}_{i\in I}$. We refer to such groups of transitions as *steps*. We omit the notation $i \in I$ when there is no ambiguity. We do not distinguish steps of the form $P\{\stackrel{\alpha_i}{\longmapsto}_{p_i} P_i, \stackrel{\beta}{\longmapsto}_{q_1} Q, \stackrel{\beta}{\longmapsto}_{q_2} Q\}$ and $P\{\stackrel{\alpha_i}{\longmapsto}_{p_i} P_i, \stackrel{\beta}{\longmapsto}_{q_1+q_2} Q\}$

Rules $(Cong)$, (Sum) and (Rec) are the probabilistic extension of the corresponding rules in the classical π-calculus.

$$\text{(Sum)}: \quad \frac{}{\Sigma_i (p_i,\mu_i).P_i\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}}$$

$$\text{(Cong)}: \quad \frac{P\equiv P' \quad P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\} \quad \forall i. P_i \equiv P_i'}{P'\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i'\}}$$

$$\text{(Rec)}: \quad \frac{P[rec_X P/X]\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}}{rec_X P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}}$$

The (Com) rule corresponds to the fusion of the three classical rules of the π-calculus for interleaving, communication and communication with scope extrusion (called PAR, COM and CLOSE traditionally).

$$(Com): \quad \frac{P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\} \quad Q\{\stackrel{\eta_j}{\longmapsto}_{q_j} Q_j\} \quad \forall i,j. \quad bn(\mu_i)\bigcap fn(Q_j)=\emptyset \quad \wedge \quad bn(\eta_j)\bigcap fn(P_i)=\emptyset}{(P|Q)\{\stackrel{\alpha_{i,j}}{\longmapsto}_{p_i q_j} R_{i,j}\}}$$

where $R_{i,j}$ and $\alpha_{i,j}$ are defined by:

1. if $\mu_i = \overline{y}x,\ \eta_j = y(z)$,
   (a) either $R_{i,j} = P_i|Q_j[x/z] \quad \wedge \quad \alpha_{i,j} = \tau$: communication.
   (b) or $R_{i,j} = P_i|Q \quad \wedge \quad \alpha_{i,j} = \mu_i$: left interleaving.
   (c) or $R_{i,j} = P|Q_j \quad \wedge \quad \alpha_{i,j} = \eta_j$: right interleaving.
2. symmetric case: $\mu_i = y(z),\ \eta_j = \overline{y}x$
3. if $\mu_i = \overline{y}(x),\ \eta_j = y(z)$,
   (a) either $R_{i,j} = (x)(P_i|Q_j[x/z]) \quad \wedge \quad \alpha_{i,j} = \tau$: communication and scope extrusion
   (b) or $R_{i,j} = P_i|Q \quad \wedge \quad \alpha_{i,j} = \mu_i$: left interleaving.

(c) or $R_{i,j} = P|Q_j \quad \wedge \quad \alpha_{i,j} = \eta_j$: right interleaving.

4. symmetric case: $\mu_i = y(z),\ \eta_j = \overline{y}(x)$

5. otherwise,

   (a) either $R_{i,j} = P_i|Q \quad \wedge \quad \alpha_{i,j} = \mu_i$: left interleaving.

   (b) or $R_{i,j} = P|Q_j \quad \wedge \quad \alpha_{i,j} = \eta_j$: right interleaving.

This is more complicated than other probabilistic calculi in literature because we are dealing with an asynchronous model (asynchronous in the sense of no global clock): each process can proceed at his own speed and decide whether to synchronize or not, and this for each transition of the steps, hence several different cases can occur when combining the steps of two parallel processes.

Given two steps $P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}_{i\in I}$ and $Q\{\stackrel{\eta_j}{\longmapsto}_{q_j} Q_j\}_{j\in J}$, we want to build a step from $P|Q$. To this end, for each pair of transitions $(P \stackrel{\mu_i}{\longmapsto}_{p_i} P_i, Q \stackrel{\eta_j}{\longmapsto}_{q_j} Q_j)$ we build a transition of $P|Q$ using one of the three classical rules for the parallel composition. For instance, if $P\{\stackrel{\overline{x}y}{\longmapsto}_{1/2} P', \ldots\}$ and $Q\{\stackrel{x(z)}{\longmapsto}_{1/3} Q', \ldots\}$, then from $P|Q$ we will have steps of the form $P|Q\{\stackrel{\tau}{\longmapsto}_{1/6} P'|Q'[y/z], \ldots\}$ (communication), of the form $P|Q\{\stackrel{\overline{x}y}{\longmapsto}_{1/6} P'|Q, \ldots\}$ (left interleaving), and of the form $P|Q\{\stackrel{x(z)}{\longmapsto}_{1/6} P|Q', \ldots\}$ (right interleaving).

One may wonder why we do not put these steps together in one single step from $P|Q$. This is because the alternative between these three cases should be non-deterministic rather than probabilistic, as in the classical π-calculus.

Note that the non-determinism of the calculus derives from this rule only.

Let us illustrate the rule COM with an example. For simplicity we omit the parameters in the communication. Furthermore, if in a step we have two transition with the same label and the same continuation, then we write the transition only once, of course with probability equal to the sum of the probabilities.

Example 6.6.1 *Consider the processes* $P = (1/2, \overline{y}).P_1 + (1/2, x).P_2$ *and* $Q = (1/3, y).Q_1 + (2/3, z).Q_2$. *The possible steps of* $P \mid Q$ *are* 24, *in fact* 3 *possible outcomes derive from the combination between the first branch of* P *and the first of* Q, 2 *from the first of* P *and the second of* Q, 2 *from the second of* P *and the first of* Q, *and* 2 *from the second of* P *and the second of* Q. *All the possible*

steps are :

$$P\,|\,Q\{\stackrel{\tau}{\longmapsto}_{1/6} P_1\,|\,Q_1,\quad \stackrel{y}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/2} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\tau}{\longmapsto}_{1/6} P_1\,|\,Q_1,\quad \stackrel{y}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/6} P_2\,|\,Q,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2\}$$
$$P\,|\,Q\{\stackrel{\tau}{\longmapsto}_{1/6} P_1\,|\,Q_1,\quad \stackrel{y}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/3} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\tau}{\longmapsto}_{1/6} P_1\,|\,Q_1,\quad \stackrel{\overline{y}}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2\}$$
$$P\,|\,Q\{\stackrel{\tau}{\longmapsto}_{1/6} P_1\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2,\quad \stackrel{x}{\longmapsto}_{1/2} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\tau}{\longmapsto}_{1/6} P_1\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{2/3} P\,|\,Q_2,\quad \stackrel{x}{\longmapsto}_{1/6} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\tau}{\longmapsto}_{1/6} P_1\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2,\quad \stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/3} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\tau}{\longmapsto}_{1/6} P_1\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{2/3} P\,|\,Q_2,\quad \stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1\}$$
$$P\,|\,Q\{\stackrel{\overline{y}}{\longmapsto}_{1/2} P_1\,|\,Q,\quad \stackrel{x}{\longmapsto}_{1/2} P_2\,|\,Q\}$$

$$P\,|\,Q\{\stackrel{\overline{y}}{\longmapsto}_{1/2} P_1\,|\,Q,\quad \stackrel{x}{\longmapsto}_{1/6} P_2\,|\,Q,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2\}$$
$$P\,|\,Q\{\stackrel{\overline{y}}{\longmapsto}_{1/2} P_1\,|\,Q,\quad \stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/3} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\overline{y}}{\longmapsto}_{1/2} P_1\,|\,Q,\quad \stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2\}$$
$$P\,|\,Q\{\stackrel{\overline{y}}{\longmapsto}_{1/6} P_1\,|\,Q,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2,\quad \stackrel{x}{\longmapsto}_{1/2} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\overline{y}}{\longmapsto}_{1/6} P_1\,|\,Q,\quad \stackrel{z}{\longmapsto}_{2/3} P\,|\,Q_2,\quad \stackrel{x}{\longmapsto}_{1/6} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\overline{y}}{\longmapsto}_{1/6} P_1\,|\,Q,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2,\quad \stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/3} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{\overline{y}}{\longmapsto}_{1/6} P_1\,|\,Q,\quad \stackrel{z}{\longmapsto}_{2/3} P\,|\,Q_2,\quad \stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1\}$$
$$P\,|\,Q\{\stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{\overline{y}}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/2} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{\overline{y}}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/6} P_2\,|\,Q,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2\}$$
$$P\,|\,Q\{\stackrel{y}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{\overline{y}}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{x}{\longmapsto}_{1/3} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{y}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{\overline{y}}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2\}$$
$$P\,|\,Q\{\stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2,\quad \stackrel{x}{\longmapsto}_{1/2} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{y}{\longmapsto}_{1/6} P\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{2/3} P\,|\,Q_2,\quad \stackrel{x}{\longmapsto}_{1/6} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{y}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{1/3} P\,|\,Q_2,\quad \stackrel{x}{\longmapsto}_{1/3} P_2\,|\,Q\}$$
$$P\,|\,Q\{\stackrel{y}{\longmapsto}_{1/3} P\,|\,Q_1,\quad \stackrel{z}{\longmapsto}_{2/3} P\,|\,Q_2\}$$

The two (New) rules. The four previous rules are the same for both semantics. But as restrictions can erase some actions (if $x \in fn(\mu) \wedge \neg(\mu = \overline{z}x \wedge z \neq x)$, then (x) prevents P from doing μ) one has, in the probabilistic case, to renormalize coefficients so that the sum stays equal to 1. In the quantitative case it is sufficient to erase transitions which do not satisfy the condition without modifying coefficients.

As for the rule (Com), each of these two rules correspond to several rules of the classical π-calculus. In each rule, the first set corresponds to the actions that output the name bound by the (x). The second set corresponds to the action that do not involve any name bound by the (x). These cases corresponds to the classic rules OPEN and NU.

The quantitative case.

$$(New_q): \quad \frac{P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\} \quad \exists i.(x \notin fn(\mu_i) \vee (\mu_i = \overline{z}x \wedge z \neq x))}{(x)P\{\stackrel{\overline{z_i}(x)}{\longmapsto}_{p_i} P_i, \quad \mu_i = \overline{z_i}x, z_i \neq x\} \bigcup \{\stackrel{\mu_i}{\longmapsto}_{p_i} (x)P_i, \quad x \notin fn(\mu_i)\}}$$

The probabilistic case. The rule for the probabilistic case is obtained from previous one by re-normalizing the coefficients.

$$(New_p): \quad \frac{P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\} \quad \exists i.(x \notin fn(\mu_i) \vee (\mu_i = \overline{z}x \wedge z \neq x))}{(x)P\{\stackrel{\overline{z_i}(x)}{\longmapsto}_{q_i} P_i, \quad \mu_i = \overline{z_i}x, z_i \neq x\} \bigcup \{\stackrel{\mu_i}{\longmapsto}_{q_i} (x)P_i, \quad x \notin fn(\mu_i)\}}$$

with $\forall i.q_i = p_i/(\Sigma_{j:x \notin fn(\mu_j) \vee (\mu_j = \overline{z}x \wedge z \neq x)} p_j)$

Note that each rule, except New_q for the quantitative case, preserves the sum of the coefficients. Thus in the probabilistic case we derive only steps where the sum of the coefficients is equal to 1.

Note also that the renormalization in (New_p) is the aspect of the probabilistic π-calculus that we do not manage to encode in the next section.

6.6.2 Weak steps.

The weak steps are defined by:

$$\mathsf{wea1}: \quad \frac{P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}}{P\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}}$$

$$\mathsf{wea2}: \quad \frac{P\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\} \uplus \{\stackrel{\tau}{\Longrightarrow}_q Q\} \quad Q\{\stackrel{\eta_j}{\Longrightarrow}_{r_j} R_j\}}{P\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\} \uplus \{\stackrel{\eta_j}{\Longrightarrow}_{q.r_j} R_j\}}$$

$$\mathsf{wea3}: \quad \frac{\forall n.P\{\stackrel{\mu_i}{\Longrightarrow}_{p_{i\,n}} P_i\} \quad lim_{n \to \infty} p_{i\,n} = \overline{p_i}}{P\{\stackrel{\mu_i}{\Longrightarrow}_{\overline{p_i}} P_i\}}$$

The first two rules are inspired by [31]. The last one is new and represents the limit case of the second one. Note that the limit only concerns probabilities.This last rule is very important because our encoding is based on a loop that allows one to backtrack whenever we take the wrong decision. Eventually, the right decision will be taken with probability 1 but this may happen only in the limit.

Note that also here the sum of the coefficients is preserved.

6.7 Encoding the mixed choice into the separate choice.

6.7.1 The encoding.

We show that, in the probabilistic π-calculus without name restriction and in the quantitative π-calculus, the mixed choice and the separate one are equally expressive. The reason why we do not consider the restriction operator is because, at present, we are not able to provide a correct probabilistic encoding.

The correctness of the encoding is stated in Theorems 6.7.1 and 6.7.2 for the probabilistic semantics and in Theorems 6.7.3 and 6.7.4 for the quantitative semantics.

Definition 6.7.1 *The encoding $[\![.]\!]$ is defined by:*

$$\begin{aligned} [\![(x)P]\!] &= (x)[\![P]\!] \\ [\![P|Q]\!] &= [\![P]\!]|[\![Q]\!] \\ [\![rex_X.P]\!] &= rec_X.[\![P]\!] \\ [\![X]\!] &= X \end{aligned}$$

Let P be a mixed choice of the form $\Sigma_{i\in I}(p_i, \overline{x_i}y_i).P_i + \Sigma_{j\in J}(q_j, \tau).Q_j + \Sigma_{k\in K}(r_k, x_k(z_k)).R_k$. For W defined to be $\Sigma_{i\in I}\ p_i + \Sigma_{j\in J}\ q_j + \Sigma_{k\in K}\ r_k$, we define the encoding of P as follows:

$$\begin{aligned} [\![P]\!] = \quad rec_X.(\quad & (\frac{\Sigma_{i\in I}\ p_i}{W}, \tau).P_{send} \\ & +(\frac{\Sigma_{j\in J}\ q_j}{W}, \tau).P_\tau \\ & +(\frac{\Sigma_{k\in K}\ r_k}{W}, \tau).P_{receive} \\ ) & \end{aligned}$$

where:

$$\begin{aligned} P_{send} = \quad & \Sigma_{i\in I}(\frac{W\times(1-\epsilon)}{\Sigma_{i\in I}(p_i)}p_i, \overline{x_i}y_i).[\![P_i]\!] + (\epsilon, \tau).X \\ P_\tau = \quad & \Sigma_{j\in J}(\frac{W\times(1-\epsilon)}{\Sigma_{j\in J}(q_j)}q_j, \tau).[\![Q_j]\!] + (\epsilon, \tau).X \\ P_{receive} = \quad & \Sigma_{k\in K}(\frac{W\times(1-\epsilon)}{\Sigma_{k\in K}(r_k)}r_k, x_k(z_k)).[\![R_k]\!] + (\epsilon, \tau).X \end{aligned}$$

Our encoding is homomorphic w.r.t. all constructors except the choice. Let P be a mixed choice $\Sigma_{i\in I}(p_i, \overline{x_i}y_i).P_i + \Sigma_{j\in J}(q_j, \tau).Q_j + \Sigma_{k\in K}(r_k, x_k(z_k)).R_k$. We begin by making a blind choice (that is guarded by τ-actions) between three branches corresponding to outputs, inputs or τ's. Then in each branch we make a (separate) choice between the outputs, inputs or τ. As one can go in the outputs branch (for instance) even if the context enforces communication on an input, we have to include a mechanism to backtrack. To this end, the separate choice contains also a τ-prefixed branch going back recursively to the beginning, with weight ϵ, which needs to be smaller than 1. This means that the process can, in principle, loop forever, however the probability of this event is 0.

6.7.2 Correctness of the encoding.

The structure of the proof.

Completeness of the encoding. Theorems 6.7.1 and 6.7.3 state that if P can perform a step, then $[\![P]\!]$ can perform the corresponding weak step. To prove these results, we use Lemma 6.7.1. Essentially, the properties stated by the items 1 to 5 of this lemma show that the weak variants of Rules (Sum), (Rec), (Cong), (Com) and New_q, respectively, are sound with respect to $\Longrightarrow$. By "weak variant" of rule X here we mean the rule obtained by replacing $\longmapsto$ with $\Longrightarrow$ in X.

Lemma 6.7.1

1. *If $Q\{\stackrel{\eta_j}{\longmapsto}_{q_j} Q_j\}$ has been obtained with the (Sum) rule then $Q\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\}$,*
2. *If $P[rec_X P/X]\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$, then $rec_X P\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$,*
3. *If $P \equiv P'$, $P\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$ and $\forall i. P_i \equiv P'_i$, then $P'\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P'_i\}$,*
4. *If $P\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$, $Q\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\}$, $\forall i, j. bn(\mu_i) \bigcap fn(Q_j) = \emptyset$ and $bn(\eta_j) \bigcap fn(P_i) = \emptyset$, then $(P|Q)\{\stackrel{\alpha_{i,j}}{\Longrightarrow}_{p_i q_j} R_{i,j}\}$, where the $\alpha_{i,j}$'s and the $R_{i,j}$'s are defined as in the COM rule,*
5. *In the probabilistic semantics if $P\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$ and $\exists i.(x \notin fn(\mu_i) \vee (\mu_i = \overline{z}x \wedge z \neq x))$, then $(x)P\{\stackrel{\overline{z_i}(x)}{\Longrightarrow}_{p_i} P_i,\ \mu_i = \overline{z_i}x, z_i \neq x\} \bigcup \{\stackrel{\mu_i}{\Longrightarrow}_{p_i} (x)P_i,\ x \notin fn(\mu_i)\}$.*

Proof (of Lemma 6.7.1) The five items of the lemma are proved by a straightforward induction on the proof tree of the premise. We detail it only for the second item, the other case are identical.

- If $P[rec_X P/X]\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$ has been obtained by the rule (wea1), then $P[rec_X P/X]\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$. By application of the (Rec) rule we obtain that $rec_X P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$. Finally by application of the rule (wea1) we get $rec_X P$ $\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$.

- If $P[rec_X P/X]\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$ has been obtained by the rule (wea2), then the step can be written $P[rec_X P/X]\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\} \bigcup \{\stackrel{\rho_k}{\Longrightarrow}_{q.r_k} R_k\}$ where there is a process Q such that $P[rec_X P/X]\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\} \bigcup \{\stackrel{\tau}{\Longrightarrow}_{q} Q\}$ and $Q\{\stackrel{\rho_k}{\Longrightarrow}_{r_k} R_k\}$. By induction hypothesis we obtain that $rec_X P\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\} \bigcup \{\stackrel{\tau}{\Longrightarrow}_{q} Q\}$. And finally by the (wea2) rule $rec_X P\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\} \bigcup \{\stackrel{\rho_k}{\Longrightarrow}_{q.r_k} R_k\}$.

- If $P[rec_X P/X]\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$ has been obtained by the rule (wea2), then for all n there are $p_{i,n}$ such that $P[rec_X P/X]\{\stackrel{\mu_i}{\Longrightarrow}_{p_{i,n}} P_i\}$ and $lim_{n\to\infty} p_{i,n} = p_i$. By induction hypothesis we get $rec_X P\{\stackrel{\mu_i}{\Longrightarrow}_{p_{i,n}} P_i\}$. Finally by the (wea3) rule we obtain $rec_X P\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} P_i\}$.

□

Unfortunately the same result does not hold for the rule New_p. The following is a counterexample.

Example 6.7.1 *Let $P = (1/2, c).0 + (1/2, \tau).Q$ and $Q = (1/2, b).0 + (1/2, a).0$. We have $P\{\stackrel{c}{\longmapsto}_{1/2} 0, \stackrel{b}{\Longrightarrow}_{1/4} 0, \stackrel{a}{\Longrightarrow}_{1/4} 0\}$. If the equivalent of Lemma 6.7.1.v for New_p were to hold, then we should have also $(a)P\{\stackrel{c}{\Longrightarrow}_{2/3} 0, \stackrel{b}{\Longrightarrow}_{1/3} 0\}$ (the coefficient are different from the New_q case because we need to apply renormalization). However, we have only the strong steps $(a)P\{\stackrel{c}{\longmapsto}_{1/2} 0, \stackrel{\tau}{\longmapsto}_{1/2} \nu a.Q\}$ and $(a)Q\{\stackrel{b}{\longmapsto}_{1} 0\}$, so by Rule* wea2 *we can only obtain $(a.P)\{\stackrel{c}{\Longrightarrow}_{1/2} 0, \stackrel{b}{\Longrightarrow}_{1/2} 0\}$.*

The discrepancy illustrated by the above counterexample is due to the fact that the renormalization in the weak variant of Rules (Sum), (Rec), (Cong) and (Com) would take place at a different time than in Rules weak1-weak3.

This also implies that the homomorphic translation of the name restriction is not correct.

Soundness of the encoding. Conversely, the soundness of the encoding is stated in Theorems 6.7.2 and 6.7.4. We would like to have that if $[\![P]\!]$ can perform a step, then P can perform the corresponding weak step. However we cannot get this result: since the translation divides a mixed choice into various separate choices, we get more possibilities in the translated term than in the original.

Indeed, as a translated term can only make a blind choice, to get a standard bisimulation we would need that the P_{send}, P_{receive} and P_τ resulting from the encoding are associated with P by the bisimulation. This does not work, one can easily see that these terms have in general steps different from those of the initial term.

However, all the weak steps that $[\![P]\!]$ can perform can be continued so as to get a step in the ones of P. For instance, after $[\![P]\!]$ has performed the blind choice, it can perform the output choices, which will result in a weak step of the form $[\![P]\!]\{\stackrel{\overline{x}_i y_i}{\Longrightarrow}_{p_i} [\![P_i]\!]\} \uplus \{\stackrel{\tau}{\longmapsto}_{\Sigma_j q_j} P_\tau\} \uplus \{\stackrel{\tau}{\longmapsto}_{\Sigma_k r_k} P_{\text{receive}}\}$. By repeating this for the other two kinds of branches, we will get a weak step of the form

$[\![P]\!]\{\stackrel{\overline{x_i}y_i}{\Longrightarrow}_{p_i} P_i\} \uplus \{\stackrel{\tau}{\Longrightarrow}_{q_j} Q_j\} \uplus \{\stackrel{x_k(z_k)}{\Longrightarrow}_{r_k} R_k\}$, which corresponds exactly to the step of P.

This situation is well known in the classical setting: it is often the case that an encoding does not preserve the operational semantics at each step. In other words, it may happen that some intermediate states in the computation of an encoded process do not correspond to the encoding of any derivative of the original process. However, it is often the case that the encoding satisfies a property of the following form: if $[\![P]\!] \stackrel{\mu}{\Longrightarrow} Q$, then there exists P' such that $[\![P]\!] \stackrel{\mu}{\Longrightarrow} Q \stackrel{\tau}{\Longrightarrow} [\![P']\!]$ and $P \stackrel{\mu}{\Longrightarrow} P'$.

To formalize the above idea in the probabilistic setting, we introduce the notion of *completion* of a step:

Definition 6.7.2 *Let P be a process and let $P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$ be one of its steps. A completion of this step is a weak step obtained by first applying* wea1 *to $P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$, and then continuing with applications of* wea2 *and* wea3.

Intuitively, completing a step means to explore the possible continuations of this step, going deeper each time we get a τ. The wea3 rule indeed only modifies coefficients and the rule wea2 permits to extend only silent transitions. Thus the completion of a step does not go further, in each branch, than the first non-silent action.

We remark that the notion of completion is quite robust, in the sense that it does not reduce the interaction possibilities, as shown by the following proposition.

Proposition 6.7.1 *Let P be a process, $P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$ be one of its steps, and $P\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\}$ be one of its completions. Consider now a process R, and $R\{\stackrel{\alpha_k}{\longmapsto}_{r_k} R_k\}$ one of its steps. Let X be a step obtained by applying the* COM *rule to $P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$ and $R\{\stackrel{\alpha_k}{\longmapsto}_{r_k} R_k\}$. By applying the 4-th item of Lemma 6.7.1 with premises $P\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\}$ and $R\{\stackrel{\alpha_k}{\longmapsto}_{r_k} R_k\}$, we obtain a step Y of $P \mid R$ that is a completion of X.*

Proof The result is obtained by a straightforward induction on the proof tree of $P\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\}$. □

We are now ready to complete the formal assessment of the correctness of the encodings, first for the probabilistic semantics and then for the quantitative one.

Theorem 6.7.1 *In the probabilistic semantics, if $P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$ can be derived without using the rule New_p, then $[\![P]\!]\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} [\![P]\!]_i\}$.*

Proof The result is obtained by a standard induction on the proof tree of $P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$.

- If the transition has been obtained by a (Sum) rule, then it means that P is written $\sum_{i\in I}(p_i, \overline{x_i}\, y_i).P_i + \sum_{j\in J}(q_j, \tau).Q_j + \sum_{k\in K}(r_k, \overline{x_k}\, y_k).R_k$ and that the transition is written $P\{\stackrel{\overline{x_i}\, y_i}{\longmapsto}_{p_i}\ P_i\} \bigcup \{\stackrel{\tau}{\longmapsto}_{q_j}\ Q_j\} \bigcup \{\stackrel{\overline{x_k}\, y_k}{\longmapsto}_{r_k}\ R_k\}$. By definition of the encoding, for $W \triangleq \Sigma_{i\in I}\ p_i + \Sigma_{j\in J}\ q_j + \Sigma_{k\in K}\ r_k$:

$$\begin{aligned}[\![P]\!] = \quad rec_X.(\quad & (\frac{\Sigma_{i\in I}\ p_i}{W}, \tau).P_{\text{send}} \\ & +(\frac{\Sigma_{j\in J}\ q_j}{W}, \tau).P_\tau \\ & +(\frac{\Sigma_{k\in K}\ r_k}{W}, \tau).P_{\text{receive}} \\ &)\end{aligned}$$

where:

$$\begin{aligned} P_{\text{send}} = \quad & \Sigma_{i\in I}(\frac{W\times(1-\epsilon)}{\Sigma_{i\in I}(p_i)} p_i, \overline{x_i} y_i).[\![P_i]\!] + (\epsilon, \tau).X \\ P_\tau = \quad & \Sigma_{j\in J}(\frac{W\times(1-\epsilon)}{\Sigma_{j\in J}(q_j)} q_j, \tau).[\![Q_j]\!] + (\epsilon, \tau).X \\ P_{\text{receive}} = \quad & \Sigma_{k\in K}(\frac{W\times(1-\epsilon)}{\Sigma_{k\in K}(r_k)} r_k, x_k(z_k)).[\![R_k]\!] + (\epsilon, \tau).X \end{aligned}$$

By application of the (Sum) rule we get $[\![P]\!]\{\stackrel{\tau}{\longmapsto}_{\Sigma_{i\in I}\ p_i/W}\ P_{send}, \stackrel{\tau}{\longmapsto}_{\Sigma_{j\in J}\ q_j/W} P_\tau, \stackrel{\tau}{\longmapsto}_{\Sigma_{k\in K}\ r_k/W}\ P_{receive}\}$. By application of the (wea2) rule, we get $[\![P]\!]\{\stackrel{\overline{x_i}\, y_i}{\Longrightarrow}_{(1-\epsilon)p_i}\ [\![P_i]\!]\} \bigcup \{\stackrel{\tau}{\Longrightarrow}_{(1-\epsilon)q_j}\ [\![Q_j]\!]\} \bigcup \{\stackrel{\overline{x_k}\, y_k}{\Longrightarrow}_{(1-\epsilon)r_k}\ [\![R_k]\!]\} \bigcup \{\stackrel{\tau}{\Longrightarrow}_{\epsilon}\ [\![P]\!]\}$. By iterating this process n times we obtain the step $[\![P]\!]\{\stackrel{\overline{x_i}\, y_i}{\Longrightarrow}_{E_n(1-\epsilon)p_i}\ [\![P_i]\!]\} \bigcup \{\stackrel{\tau}{\Longrightarrow}_{E_n(1-\epsilon)q_j}\ [\![Q_j]\!]\} \bigcup \{\stackrel{\overline{x_k}\, y_k}{\Longrightarrow}_{E_n(1-\epsilon)r_k}\ [\![R_k]\!]\} \bigcup \{\stackrel{\tau}{\Longrightarrow}_{\epsilon^n}\ [\![P]\!]\}$, where $E_n = \sum_{1\le i\le n} \epsilon^i$. Since $lim_{n\to\infty}(1-\epsilon)\times E_n = \frac{(1-\epsilon)}{(1-\epsilon)} = 1$, and $lim_{n\to\infty}\epsilon^n = 0$ because $\epsilon \le 1$, we can apply the (wea3) rule and get $[\![P]\!]\{\stackrel{\overline{x_i}\, y_i}{\Longrightarrow}_{p_i}\ [\![P_i]\!]\} \bigcup \{\stackrel{\tau}{\Longrightarrow}_{q_j}\ [\![Q_j]\!]\} \bigcup \{\stackrel{\overline{x_k}\, y_k}{\Longrightarrow}_{r_k}\ [\![R_k]\!]\}$.

- If the transition has been obtained by a rule (Rec), (Cong) or (Com), the result is easily obtained by the induction hypothesis and the items 2, 3 and 4 of Lemma 6.7.1 respectively.

□

Theorem 6.7.2 *In the probabilistic semantics, if $[\![P]\!]\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$ can be derived without using the (New_p) rule, then there exist Q_j's such that $[\![P]\!]\{\stackrel{\eta_j}{\Longrightarrow}_{q_j}\ [\![Q_j]\!]\}$ is a derivable completion of the above step and $P\{\stackrel{\eta_j}{\Longrightarrow}_{q_j}\ Q_j\}$.*

Proof The result is obtained by a standard induction on P.

- If P is a choice written $\sum_{i\in I}(p_i, \overline{x_i}\, y_i).P_i + \sum_{j\in J}(q_j, \tau).Q_j + \sum_{k\in K}(r_k, \overline{x_k}\, y_k).R_k$. By definition of the encoding, for $W \triangleq \Sigma_{i\in I}\ p_i + \Sigma_{j\in J}\ q_j + \Sigma_{k\in K}\ r_k$:

$$\begin{aligned}\llbracket P \rrbracket = \quad rec_X.(\quad & (\frac{\Sigma_{i\in I}\ p_i}{W}, \tau).P_{\text{send}} \\ & +(\frac{\Sigma_{j\in J}\ q_j}{W}, \tau).P_\tau \\ & +(\frac{\Sigma_{k\in K}\ r_k}{W}, \tau).P_{\text{receive}} \\ &)\end{aligned}$$

where:

$$\begin{aligned} P_{\text{send}} = \quad & \Sigma_{i\in I}(\frac{W\times(1-\epsilon)}{\Sigma_{i\in I}(p_i)}p_i, \overline{x_i}y_i).\llbracket P_i \rrbracket + (\epsilon, \tau).X \\ P_\tau = \quad & \Sigma_{j\in J}(\frac{W\times(1-\epsilon)}{\Sigma_{j\in J}(q_j)}q_j, \tau).\llbracket Q_j \rrbracket + (\epsilon, \tau).X \\ P_{\text{receive}} = \quad & \Sigma_{k\in K}(\frac{W\times(1-\epsilon)}{\Sigma_{k\in K}(r_k)}r_k, x_k(z_k)).\llbracket R_k \rrbracket + (\epsilon, \tau).X \end{aligned}$$

So the only step of transitions of P and $\llbracket P \rrbracket$ are respectively $P\{\overset{\overline{x_i}\, y_i}{\longmapsto}_{p_i} P_i\} \bigcup \{\overset{\tau}{\longmapsto}_{q_j} Q_j\} \bigcup \{\overset{\overline{x_k}\, y_k}{\longmapsto}_{r_k} R_k\}$ and $\llbracket P \rrbracket \{\overset{\tau}{\longmapsto}_{\Sigma_{i\in I}\ p_i/W} P_{send},\ \overset{\tau}{\longmapsto}_{\Sigma_{j\in J}\ q_j/W} P_\tau,\ \overset{\tau}{\longmapsto}_{\Sigma_{k\in K}\ r_k/W} P_{receive}\}$. So we prove that the step of $\llbracket P \rrbracket$ has the following completion: $\llbracket P \rrbracket \{\overset{\overline{x_i}\, y_i}{\Longrightarrow}_{p_i} \llbracket P_i \rrbracket\} \bigcup \{\overset{\tau}{\Longrightarrow}_{q_j} \llbracket Q_j \rrbracket\} \bigcup \{\overset{\overline{x_k}\, y_k}{\Longrightarrow}_{r_k} \llbracket R_k \rrbracket\}$.

By application of the (wea2) rule, we get $\llbracket P \rrbracket \{\overset{\overline{x_i}\, y_i}{\Longrightarrow}_{(1-\epsilon)p_i} \llbracket P_i \rrbracket\} \bigcup \{\overset{\tau}{\Longrightarrow}_{(1-\epsilon)q_j} \llbracket Q_j \rrbracket\} \bigcup \{\overset{\overline{x_k}\, y_k}{\Longrightarrow}_{(1-\epsilon)r_k} \llbracket R_k \rrbracket\} \bigcup \{\overset{\tau}{\Longrightarrow}_{\epsilon} X\}$. By iterating this process n times we obtain the step $\llbracket P \rrbracket \{\overset{\overline{x_i}\, y_i}{\Longrightarrow}_{(1-\epsilon)\times E_n p_i} \llbracket P_i \rrbracket\} \bigcup \{\overset{\tau}{\Longrightarrow}_{(1-\epsilon)\times E_n q_j} \llbracket Q_j \rrbracket\} \bigcup \{\overset{\overline{x_k}\, y_k}{\Longrightarrow}_{(1-\epsilon)\times E_n r_k} \llbracket R_k \rrbracket\} \bigcup \{\overset{\tau}{\Longrightarrow}_{\epsilon^n} X\}$, where $E_n = \sum_{1\le i\le n} \epsilon^i$. Since $lim_{n\to\infty}(1-\epsilon)\times E_n = \frac{(1-\epsilon)}{(1-\epsilon)} = 1$, and $lim_{n\to\infty}\epsilon^n = 0$ because $\epsilon \le 1$, we can apply the (wea3) rule and get $\llbracket P \rrbracket \{\overset{\overline{x_i}\, y_i}{\Longrightarrow}_{p_i} \llbracket P_i \rrbracket\} \bigcup \{\overset{\tau}{\Longrightarrow}_{q_j} \llbracket Q_j \rrbracket\} \bigcup \{\overset{\overline{x_k}\, y_k}{\Longrightarrow}_{r_k} \llbracket R_k \rrbracket\}$.

- Otherwise the result is easily obtained using the induction hypothesis, by the items 2, 3 and 4 of Lemma 6.7.1 and by the fact that the encoding is uniform on every operator but the choice.

□

Theorem 6.7.3 *In the quantitative semantics, if* $P\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$, *then* $[\![P]\!]\{\stackrel{\mu_i}{\Longrightarrow}_{p_i} [\![P]\!]_i\}$.

Proof The result is achieved using the same arguments as for Theorem 6.7.1.□

Theorem 6.7.4 *In the quantitative semantics, if* $[\![P]\!]\{\stackrel{\mu_i}{\longmapsto}_{p_i} P_i\}$, *then there exist* Q_j*'s such that* $[\![P]\!]\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} [\![Q_j]\!]\}$ *is a derivable completion of the above step and* $P\{\stackrel{\eta_j}{\Longrightarrow}_{q_j} Q_j\}$.

Proof The result is achieved using the same arguments than for Theorem 6.7.3. □

We conclude this section with the following remark.

Remarque 7 *We can use the same mechanism to reduce the size of separate choice to two. This reduces the language to a very simple form of separate choices: blind choices, and choices of size 2 in which one of the branches is prefixed by a* τ. *The encoding works exactly in the same way as the previous one. We separate each branch of the separate choice, as we separated inputs from outputs and from* τ. *The correctness theorems are identical.*

$$
\begin{array}{rcl}
[\![\Sigma_{i\in I}(p_i,\mu_i).P_i]\!] & = & \\
 & rec_X.(& \\
 & & \Sigma_{i\in I} \quad (p_i,\tau).Q_i \\
 &) & \\
\textit{where: } Q_i & = & \qquad (1-\epsilon,\mu_i).[\![P_i]\!] + (\epsilon,\tau).X
\end{array}
$$

6.8 Expressiveness of the separate choice.

We just showed how the mixed choice can be reduced to separate choice of size two. The question is now the relative expressiveness of the input-guarded and output-guarded choice. First we simply show how to lift to the probabilistic settings the classical encodings of [44] between output guarded choice and input guarded choice:

$$
\begin{array}{rcl}
[\![x(y).P]\!] & = & (z)(\overline{x}z \quad | \quad z(y).[\![P]\!]) \\
[\![\overline{x}y.Q]\!] & = & x(z).(\overline{z}y \quad | \quad [\![Q]\!])
\end{array}
$$

And we make the conjecture that there is no encoding from the pair of separate choice to only one of them.

6.8.1 Encodings between input and output guarded choices.

These encodings highlight the symmetry between the two choices and establish that they are equally expressive. One has to note that the encodings only use asynchronous outputs (although they can be used within choices). So all the four languages with either input or output choice, and with asynchronous output or not, are equivalent.

Encoding the input guarded choice into the output guarded choice.

$$[\![\Sigma_{i\in I}(p_i, x_i(y_i)).P_i]\!] \quad = \quad (z_i)_{i\in I}(\quad \Sigma_{i\in I}(p_i, \overline{x_i}z_i) \quad | \quad \Pi_{i\in I} z_i(y_i).[\![P_i]\!] \quad)$$

$$[\![\overline{x}y.P]\!] \quad = \quad x(z).(\overline{z}y \quad | \quad [\![P]\!])$$

Encoding the output guarded choice into the input guarded choice.

$$[\![x(y).P]\!] \quad = \quad (z)(\overline{x}z|z(y).[\![P]\!])$$

$$[\![\Sigma_{i\in I}(p_i, \overline{x_i}y_i).P_i]\!] \quad = \quad \Sigma_{i\in I}(p_i, x_i(z)).(\overline{z}y_i|[\![P_i]\!])$$

6.8.2 A failed attempt to encode the separate choice.

We present here our best attempt to encode separate choice by input guarded choices, we discuss the reason why it does not work, and we conjecture that it is not possible to define such an encoding.

In a non probabilistic setting, various kinds of choice have been encoded by using the parallel operator. The basic idea is to put in parallel the branches of the choice, making sure that only one of them would be executed. See for instance [57, 58]. This idea however cannot work here, since the transitions would not be in the same step anymore. In other words, we cannot encode choice with parallelism since in the choice the decision between two branches is ruled by probabilities, while in the parallel product it is ruled by non-determinism. So a choice can only be translated in another choice, similar for actions and probabilities.

So, our only hope is to translate separate choices into input guarded choices. Let us, for instance, consider again the idea of [44]. This encoding can be lifted to choices in the following way:

$$
\begin{array}{lll}
[\![(p, x(y)).P_x + (1-p, \tau).P_\tau]\!] & = & \\
 & (z)(& \\
 & \overline{x}z & \mid \quad (p, z(y)).[\![P_x]\!] + (1-p, \tau).[\![P_\tau]\!] \\
 &) &
\end{array}
$$

$$
\begin{array}{lll}
[\![(p_x, \overline{x}y).P_x + (1-p_x, \tau).P_\tau]\!] & = & \\
 & (p_x, x(z)).(\quad \overline{z}y | [\![P_x]\!] \quad) & \\
 & + \quad (1-p_x, \tau).[\![P_\tau]\!] &
\end{array}
$$

However this does not work since the translation of the output guarded choice can synchronize with $\overline{x}z$ while simultaneously the translation of the input guarded choice can execute its τ. The input choice performed its τ but the output choice began its branch of synchronization, and there is no way to go backwards. As we do not have output choices, the backward mechanism of the previous encoding can not work for the outputs. The translation of the outputs guarded choice is now in a deadlock which was not possible in the original term.

A solution could be to replace the τ of the translation of the input choice by an input on channel x so that the $\overline{x}z$ can synchronize either with this branch or with the input branch of the translation of the outputs guarded choice. But then another input guarded choice on x can interfere. It also contains a $\overline{x}z'$ that can be received by the first input guarded choice while the $\overline{x}z$ would be received by the output guarded choice. Here again we get an unexpected deadlock.

6.9 Discussion and criticisms on the design of our calculus.

In the design of our language we made some decisions that may be controversial, in particular regarding the rule for the parallel composition. We had a discussion with Roberto Segala on this subject, and he came out with the following example:

Consider the following processes:

- $P = (1/2, \overline{a}).0 + (1/2, \overline{b}).0$
- $Q = rec_X((1/2, a).X + (1/2, c).X)$

A possible step for $P \,|\, Q$ is:

$$P \,|\, Q\{\stackrel{\tau}{\longmapsto}_{1/4} Q, \stackrel{a}{\longmapsto}_{1/4} P \,|\, Q, \stackrel{c}{\longmapsto}_{1/2} P \,|\, Q\}$$

If the communication does not occur then the system stays in the same state. Thus by scheduling always this step, we have a computation where the

communication occurs eventually with probability 1. Roberto Segala did not find this acceptable since $\overline{a}$ has probability 1/2 in P, so there should be no context where the synchronization on the channel a occurs with a probability greater than 1/2.

The key point is this example is that the decision taken are not kept in memory. Indeed if the synchronization does not occur, it could be the case that it is because P has decided to make a $\overline{b}$. But at the next step, we reschedule the same step (with probability 1/4 of synchronizing) without taking account of the fact that P had already selected $\overline{b}$.

However we believe that the intuition of the choice mechanisms is that the commitment to a certain branch in a choice takes place at the same time as the transition, so we find natural to think of the choice as memory-less, and in the above example we find it correct that the communication occurs eventually with probability 1. But this is a matter of interpretation of course, and so we leave up to the reader to decide.

6.10 Conclusion.

In the classical settings, it has been proved that the separate choice is strictly less expressive than the mixed choice. In the main results of this Chapter we have proved that, in the stochastic settings, the same expressiveness gap exists, when both calculi have or do not have infinite rates. On the contrary, we have proved that in the probabilistic settings both choices are equally expressive. This provides a panorama of the relative expressive power of the separate and mixed choices.

6.11 Related work.

We are not aware of studies of the expressiveness of the stochastic π-calculus. On the contrary our results concerning the probabilistic settings present some analogy with the one in [60], where mixed choice was encoded using probabilistic input-guarded choice. The difference is that in our case we have separate choice available, and we present a much simpler encoding, based on a default possibility of backtracking. The encoding in [60] is based on a sophisticated extension of the dining cryptographers protocol. We think that such idea cannot be extended to our setting. On the other hand, we are not sure either that our simpler encoding can be adapted somehow to the setting in [60], because ours requires a choice construct with both output and τ prefixes, which is not present in the probabilistic asynchronous π-calculus considered in [60].

Chapter 7

Conclusion.

Summary. We have presented general principles governing the design of nano-scale machines and explained why these devices are intrinsically compositional both in structure and function, as well as concurrent. These properties indicate that formalisms originating from the Concurrency theory and in particular process algebra are interesting candidates for the formal modeling, simulation and analysis of such systems. Our choice of formalism was driven by three main criterions: our language should possess a stochastic semantics, it should be rule-based and it should be able to explicitly represent the molecular bounding capabilities and changes of configuration. The κ-calculus could have been a good choice, however we preferred to introduce the $\texttt{nano}\kappa$ calculus, which is more adequate for the formal representation of nano-device.

The $\texttt{nano}\kappa$ calculus is chemical-like and simple, it provides modeling easy to compose, modify and reuse. We have illustrated our approach with a case study of a 2-rotaxane consisting of a formal modeling in the $\texttt{nano}\kappa$ calculus and a series of in-silico simulations. First we validated our model by simulating the up and down motion of the ring around the axle of the rotaxane. The curves obtained by our in-silico experiment match well with the curves measured in practice. We have also simulated the rotaxane under conditions of concentration not observable in practice, and we were able to show that a classical chemical assumption was no longer valid in this setting.

We also provided an encoding of the $\texttt{nano}\kappa$ calculus in the $\texttt{nano}\pi$-calculus. It satisfies a correctness property saying essentially that a solution S has a $\texttt{nano}\kappa$-transition of rate λ to a solution T if and only the encoding of the solution S has a $\texttt{nano}\pi$-transition of rate λ to the encoding of the solution T. This is a very strong property and we believe that our encoding should be seen as an homomorphism. Moreover this close correspondence between the two languages should permit us to benefit at the same time of much of the well-known theory and the various tools of the π-calculus (which includes $\texttt{nano}\pi$) and of the nice modeling properties of the $\texttt{nano}\kappa$ calculus.

We have established a bridge between Chemistry and Computer Science at the semantics level by highlighting the close relationship between the chemical master equation and the backward stochastic bisimulation. Indeed, we introduced an equivalence motivated only from the perspective of the chemical master equation and showed that it corresponds exactly to the backward stochastic bisimulation. Moreover, we explained why we believe that the master equation equivalence resulting from this study could be the first step toward a biochemical metric, which promises to have fruitful applications such as the reduction of the state space of our models or the sensitivity analysis.

Our journey in the world of nano-devices made also a detour by the Theoretical Computer Science. Indeed the establishing of our encoding from $\texttt{nano}\kappa$ to $\texttt{nano}\pi$ had an unexpected fallout on the expressiveness of the stochastic π-calculus. Since our encoding necessitated the mixed choice, we were able to prove that the separate choice is strictly less expressive than the mixed one in the stochastic setting. The result has been proved in presence or absence of infinite rates, and we proved also that separate choice with infinite rate can encode mixed choice without infinite rate. This set of results also motivated the

introduction of the multi-scale π-calculus, where rates can be of several order of magnitude and to which we extend our expressiveness results. Interestingly, the concept of rates ranging over several order of magnitude is also relevant to the modeling of biochemical systems.

Discussion and perspectives. Looking back at the previous chapters, we believe that we have provided formal modeling, simulation and analysis of nano-devices. The modeling aspect has been quite successful. Indeed, we highlighted various advantages of the modelings performed in the $\texttt{nano}\kappa$ calculus: the language is chemical-like and simple and models are easy to reuse or modify. We also believe that the $\texttt{nano}\kappa$ modelings are easy to compose. However this last point remains to be demonstrated, which we plan to achieve by reusing our model of the rotaxane for a model of the nano-elevator, a device built upon three rotaxanes.

The simulation aspect also has been successful. Indeed the simulations have validated the modeling of the rotaxane and they have permitted to study the system under conditions not observable in practice. In particular we have shown that under extreme conditions of concentration, the rotaxane is not as efficient as expected.

The analysis aspect, though successful, is not as complete as the previous ones. We provided an encoding from the $\texttt{nano}\kappa$ calculus to the $\texttt{nano}\pi$-calculus, which should permit us to reuse much of the tools and theory of the π-calculus, and besides we provided a bridge between chemistry and computer science at the semantics level. These results promise various analysis applications such as model-checking or abstract interpretation for instance as well as potential theoretical development in terms of metrics. However the efficiency and interest of these achievements still need to be illustrated by concrete examples. Our main goals for the future are an automatic model-checker for the $\texttt{nano}\kappa$ calculus and establishing a metric based on the master equation equivalence.

Once again miscegenation has proved itself to be valuable. The tree growing at the frontier between Chemistry and Computer Science bears numerous fruits, whose taste should please the gardeners of both worlds.

Bibliography

[1] P. R. Ashton, R. Ballardini, V. Balzani, A. Credi I. Baxter, M. C. T. Fyfe, M. T. Gandolfi, M. Gómez-López, M.-V. Martínez-Díaz, A. Piersanti, N. Spencer, J. F. Stoddart, M. Venturi, A. J. P. White, and D. J. Williams. Acid-base controllable molecular shuttles. *Journal of Am. Chem. Soc.*, 120:11932–11942, 1998.

[2] J. D. Badjic, V. Balzani, A. Credi, S. Silvi, and J. F. Stoddart. A molecular elevator. *Science*, 303:1845–1849, 2004.

[3] C. Baier, J.P. Katoen, H. Hermanns, and V. Wolf. Comparative branching-time semantics for markov chains, 2004.

[4] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, and A. Troina. Bisimulation congruences in the calculus of looping sequences. In *Proceedings of the Third International Colloquium on Theoretical Aspects of Computing (ICTAC'06)*, volume 4281 of *Lecture Notes in Computer Science*, pages 93–107. Springer, 2006.

[5] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, and A. Troina. A calculus of looping sequences for modelling microbiological systems. *Fundamenta Informaticae*, 72(1-3):21–35, 2006.

[6] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, and A. Troina. A calculus of looping sequences for modelling microbiological systems. *Fundam. Inf.*, 72(1-3):21–35, 2006.

[7] Soufiene Benkirane, Jane Hillston, Chris McCaig, Rachel Norman, and Carron Shankland. Improved continuous approximation of pepa models through epidemiological examples. *Electr. Notes Theor. Comput. Sci.*, 229(1):59–74, 2009.

[8] M. Bernardo. A survey of markovian behavioral equivalences. In *Proc. of International School on Formal Methods for the Design of Computer, Communication, and Software Systems 2007*, volume 4486 of *LNCS*, pages 180–219, 2007.

[9] Marco Bernardo and Roberto Gorrieri. Corrigendum to "a tutorial on empa: a theory of concurrent processes with nondeterminism, priorities, probabilities and time". *Theor. Comput. Sci.*, 254(1-2):691–694, 2001.

[10] Luc Bougé. On the existence of symmetric algorithms to find leaders in networks of communicating sequential processes. *Acta Inf.*, 25(2):179–201, 1988.

[11] P. Buchholz. Exact and ordinary lumpability in finite markov chains. In *J. Applied Probability*, volume 31, pages 59–74, 1994.

[12] M. Calder, S. Gilmore, and J. Hillston. Modelling the influence of rkip on the erk signalling pathway using the stochastic process algebra pepa. In *Transactions on Computational Systems Biology VII*, volume 4230 of *Lecture Notes in Computer Science*, pages 1–23, 2006.

[13] L. Calzone, F. Fages, and S. Soliman. Biocham: an environment for modeling biological systems and formalizing experimental knowledge. *Bioinformatics*, 22(14):1805–1807, 2006.

[14] L. Cardelli. On process rate semantics. *Theoretical Computer Science*, 391(3):190–215, 2008.

[15] L. Cardelli and A. Phillips. A correct abstract machine for the stochastic pi-calculus. In *Proc. of Workshop on Concurrent Models in Molecular Biology*, 2004.

[16] L. Cardelli and G. Zavattaro. On the computational power of biochemistry. In *Proc. of Algebraic Biology 2008*, volume to appear of *LNCS*, 2008.

[17] Luca Cardelli. Artificial biochemistry. Microsoft Research, University of Trento, Centre for Computational and System Biology TR-08-2006, 2006.

[18] Luca Cardelli. Invited talk: A process algebra master equation. In *QEST*, pages 219–226. IEEE Computer Society, 2007.

[19] Luca Cardelli and Andrew Phillips. Spim homepage. At research.microsoft.com/~aphillip/spim/, 2006.

[20] Benoit Champin, Pierre Mobian, and Jean-Pierre Sauvage. Transition metal complexes as molecular machine prototypes. *Chemical Society Reviews*, 36(2):358–366, 2006.

[21] F. Ciocchetta and J. Hillston. Bio-pepa: An extension of the process algebra pepa for biochemical networks. *Electronic Notes in Theoretical Computer Science*, 194(3):103–117, 2008.

[22] Federica Ciocchetta and Jane Hillston. Bio-pepa: An extension of the process algebra pepa for biochemical networks. *Electr. Notes Theor. Comput. Sci.*, 194(3):103–117, 2008.

[23] Pierre-Louis Curien, Vincent Danos, Jean Krivine, and Min Zhang. Computational self-assembly. *Theor. Comput. Sci.*, 404(1-2):61–75, 2008.

[24] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling of cellular signalling. In *Proc. of International Conference on Concurrency Theory 2007*, volume 4703 of *LNCS*, pages 17–41, 2007.

[25] V. Danos, J. Feret, W. Fontana, and J. Krivine. Scalable simulation of cellular signaling networks. In *Proc. of Asian Symposium on Programming Languages and Systems 2007*, volume 4807 of *LNCS*, pages 139–157, 2007.

[26] V. Danos and C. Laneve. Formal molecular biology. *Theoretical Computer Science*, 325(1):69–110, 2004.

[27] Vincent Danos, Jérôme Feret, Walter Fontana, Russell Harmer, and Jean Krivine. Rule-based modelling, symmetries, refinements. In Jasmin Fisher, editor, *FMSB*, volume 5054 of *Lecture Notes in Computer Science*, pages 103–122. Springer, 2008.

[28] Vincent Danos, Jérôme Feret, Walter Fontana, and Jean Krivine. Abstract interpretation of cellular signalling networks. In Francesco Logozzo, Doron Peled, and Lenore D. Zuck, editors, *VMCAI*, volume 4905 of *Lecture Notes in Computer Science*, pages 83–97. Springer, 2008.

[29] Luca de Alfaro, Marta Z. Kwiatkowska, Gethin Norman, David Parker, and Roberto Segala. Symbolic model checking of probabilistic processes using mtbdds and the kronecker representation. In *TACAS*, pages 395–410, 2000.

[30] Lorenzo Dematté, Corrado Priami, and Alessandro Romanel. The blenx language: A tutorial. In Marco Bernardo, Pierpaolo Degano, and Gianluigi Zavattaro, editors, *SFM*, volume 5016 of *Lecture Notes in Computer Science*, pages 313–365. Springer, 2008.

[31] Yuxin Deng and Catuscia Palamidessi. Axiomatizations for probabilistic finite-state behaviors. In *FoSSaCS*, pages 110–124, 2005.

[32] Josee Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labelled markov processes. *Theor. Comput. Sci.*, 318(3):323–354, 2004.

[33] Josee Desharnais, Radha Jagadeesan, Vineet Gupta, and Prakash Panangaden. The metric analogue of weak bisimulation for probabilistic processes. In *LICS*, pages 413–422. IEEE Computer Society, 2002.

[34] Josee Desharnais and Prakash Panangaden. Continuous stochastic logic characterizes bisimulation of continuous-time markov processes. *J. Log. Algebr. Program.*, 56(1-2):99–115, 2003.

[35] D.A.Leigh E.R.Kay and F.Zerbetto. Synthetic molecular motors and mechanical machines. *Angewandte Chemie International Edition*, 46 Issue 1-2:72–191, 2006.

[36] S. Garaudée, S. Silvi, M. Venturi, A. Credi, A.H. Flood, and J. F. Stoddart. Shuttling dynamics in an acid-base-switchable [2]rotaxane. *ChemPhysChem*, 6:2145, 2005.

[37] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem*, 81:2340–2361, 1977.

[38] D.Parker G.Norman, C.Palamidessi and P.Wu. Model-checking probabilistic and stochastic extensions of the pi-calculus. In *IEEE Transactions on Software engineering*, 2007.

[39] John Heath, Marta Z. Kwiatkowska, Gethin Norman, David Parker, and Oksana Tymchyshyn. Probabilistic model checking of complex biological pathways. In *CMSB*, pages 32–47, 2006.

[40] T. Hérault, R. Lassaigne, F. Magniette, and S. Peyronnet. Approximate probabilistic model checking. In *Proc. 5th International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI'04)*, volume 2937 of *LNCS*. Springer, 2004.

[41] Oltea Mihaela Herescu and Catuscia Palamidessi. Probabilistic asynchronous pi-calculus. In Jerzy Tiuryn, editor, *FoSSaCS*, volume 1784 of *Lecture Notes in Computer Science*, pages 146–160. Springer, 2000.

[42] H. Hermanns. *Interactive Markov Chains: The quest for quantitative quality*, volume 2428 of *LNCS*. Springer-Verlag, Amsterdam, 2002.

[43] Jane Hillston. *A compositional approach to performance modelling*. Cambridge University Press, New York, NY, USA, 1996.

[44] Kohei Honda and Mario Tokoro. An object calculus for asynchronous communication. In Pierre America, editor, *ECOOP*, volume 512 of *Lecture Notes in Computer Science*, pages 133–147. Springer, 1991.

[45] http://www.sti.uniurb.it/bernardo/twotowers/. Twotowers.

[46] T.-J. Huang, B. Brough, C.-M. Ho, Y. Liu, A.H. Flood, P.A. Bonvallet, H.-R. Tseng, J.F. Stoddart, M. Baller, and S. Magonov. A nanomechanical device based on linear molecular motors. *Applied Physics Letters*, 85(22):5391–5393, 2004.

[47] J.Feret. Reachability analysis of biological signalling pathways by abstract interpretation. In *Proceedings of International Conference of Computational Methods in Sciences and Engineering*, volume 963(2) of *American Institute of Physics conference proceedings*, 2007.

[48] M.C. Jimenez, C. Dietrich-Buchecker, and J.-P. Sauvage. Towards synthetic molecular muscles: Contraction and stretching of a linear rotaxane dimer. *Angew. Chem. Int. Ed.*, 39(18):3284–3287, 2000.

[49] J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. D. Van Nostrand Company, 1960.

[50] Jean Krivine, Robin Milner, and Angelo Troina. Stochastic bigraphs. *Electr. Notes Theor. Comput. Sci.*, 218:73–96, 2008.

[51] C. Laneve and A.Vitale. Expressivity in the κ-family. In *Proc. of MFPS*, ENTCS, 2008.

[52] Cosimo Laneve and Fabien Tarissan. A simple calculus of protein and cells. In *Proceeding of MeCBIC'06*, volume 1677 of *ENTCS*, 2007.

[53] M.Baroncini M.Amelia and A.Credi. A simple unimolecular multiplexer/demultiplexer. *Angewandte Chemie*, 47:6240–6243, 2008.

[54] M.-V. Martínez-Díaz, N. Spencer, and J. F. Stoddart. The self-assembly of a switchable [2]rotaxane. *Angew. Chem. Int. Ed. Engl.*, 36:1904–1907, 1997.

[55] Robin Milner. *Communicating and mobile systems: the π-calculus*. Cambridge University Press, Cambridge, 1999.

[56] Robin Milner. Bigraphical reactive systems. In Kim Guldstrand Larsen and Mogens Nielsen, editors, *CONCUR*, volume 2154 of *Lecture Notes in Computer Science*, pages 16–35. Springer, 2001.

[57] Uwe Nestmann. What is a "good" encoding of guarded choice? *Inf. Comput.*, 156(1-2):287–319, 2000.

[58] Uwe Nestmann and Benjamin C. Pierce. Decoding choice encodings. *Inf. Comput.*, 163(1):1–59, 2000.

[59] Rocco De Nicola, Ugo Montanari, and Frits W. Vaandrager. Back and forth bisimulations. In Jos C. M. Baeten and Jan Willem Klop, editors, *CONCUR*, volume 458 of *Lecture Notes in Computer Science*, pages 152–165. Springer, 1990.

[60] Catuscia Palamidessi. Comparing the expressive power of the synchronous and asynchronous pi-calculi. *Mathematical Structures in Computer Science*, 13(5):685–719, 2003.

[61] Joachim Parrow. Expressiveness of process algebras. *Electr. Notes Theor. Comput. Sci.*, 209:173–186, 2008.

[62] G. D. Plotkin. A Structural Approach to Operational Semantics. Technical Report DAIMI FN-19, University of Aarhus, 1981.

[63] C. Priami and P. Quaglia. Beta binders for biological interactions. In *Proceedings of Computational Methods in Systems Biology (CMSB'04)*, volume 3082 of *Lecture Notes in Computer Science*, pages 20–33. Springer, 2005.

[64] Corrado Priami. Stochastic pi-calculus. *Computer Journal*, 38(7):578–589, 1995.

[65] Corrado Priami. Stochastic pi-calculus with general distributions. In Marina Ribando, editor, *Proceedings of PAPM '96*. CLUP, 1996.

[66] Corrado Priami, Aviv Regev, Ehud Shapiro, and William Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters*, 80:25–31, 2001.

[67] Aviv Regev, William Silverman, and Ehud Shapiro. Representation and simulation of biochemical processes using the π-calculus process algebra. In R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein, editors, *Pacific Symposium on Biocomputing*, volume 6, pages 459–470, Singapore, 2001. World Scientific Press.

[68] J.-P. Sauvage and C. O. Dietrich-Bucheker (eds.). Molecular catenanes, rotaxanes and knots. In *Wiley-VCH*, Weinheim, 1999.

[69] J. Sproston and S.Donatelli. Backward bisimulation in markov chain model checking. In *IEEE Transactions on Software Engineering*, volume 32, pages 531–546, 2006.

[70] William J. Stewart. *Introduction to the Numerical Solution of Markov Chains.* Princeton University Press., 1994.

[71] M. Ullah and O. Wolkenhauer. A general derivation of the differential Chapman-Kolmogorov equation. *eprint arXiv:q-bio/0610003*, October 2006.

[72] Frits W. Vaandrager. Expressive results for process algebras. In *REX Workshop*, pages 609–638, 1992.

[73] M.Venturi V.Balzani, A.Credi. Molecular logic circuits. *ChemPhysChem*, 4:49–59, 2002.

[74] Björn Victor and Faron Moller. The mobility workbench - a tool for the pi-calculus. In David L. Dill, editor, *CAV*, volume 818 of *Lecture Notes in Computer Science*, pages 428–440. Springer, 1994.